COGNITIVE SCIENCE
A Multidisciplinary Journal

Cognitive Science (2013) 1–29

ISSN: 0364-0213 print / 1551-6709 online
DOI: 10.1111/cogs.12030

A Computational and Empirical Investigation of Graphemes in Reading

Conrad Perry,[a] Johannes C. Ziegler,[b] Marco Zorzi[c]

[a]*Faculty of Life and Social Sciences, Swinburne University of Technology*
[b]*Aix-Marseille Université and Centre National de la Recherche Scientifique*
[c]*Dipartimento di Psicologia Generale and Center for Cognitive Science, Università di Padova*

Received 15 January 2012; received in revised form 30 May 2012; accepted 24 July 2012

Abstract

It is often assumed that graphemes are a crucial level of orthographic representation above letters. Current connectionist models of reading, however, do not address how the mapping from letters to graphemes is learned. One major challenge for computational modeling is therefore developing a model that learns this mapping and can assign the graphemes to linguistically meaningful categories such as the onset, vowel, and coda of a syllable. Here, we present a model that learns to do this in English for strings of any letter length and any number of syllables. The model is evaluated on error rates and further validated on the results of a behavioral experiment designed to examine ambiguities in the processing of graphemes. The results show that the model (a) chooses graphemes from letter strings with a high level of accuracy, even when trained on only a small portion of the English lexicon; (b) chooses a similar set of graphemes as people do in situations where different graphemes can potentially be selected; (c) predicts orthographic effects on segmentation which are found in human data; and (d) can be readily integrated into a full-blown model of multi-syllabic reading aloud such as CDP++ (Perry, Ziegler, & Zorzi, 2010). Altogether, these results suggest that the model provides a plausible hypothesis for the kind of computations that underlie the use of graphemes in skilled reading.

Keywords: Reading; Computational modeling; Graphemes; Connectionism; Orthography

1. Introduction

One of the problems with trying to build cognitive models of complex phenomena is that there are often many representational levels that can affect the flow and transformation of information (e.g., Anderson, Bothell, Douglass, Lebiere, & Qin, 2004; Levelt,

Correspondence should be sent to Conrad Perry, Faculty of Life and Social Sciences (Psychology), Swinburne University of Technology, John Street, Hawthorn, Vic. 3122, Australia. E-mail: conradperry@gmail.com

Roelofs, & Meyer, 1999; Perry, Ziegler, & Zorzi, 2007). To reduce complexity, connectionist modeling has often focused on information flow from a single level (input) to another (output; e.g., Baayen, Milin, Đurđević, Hendrix, & Marelli, 2011; Farah & McClelland, 1991; Gluck & Bower, 1988). Because of this, some sort of structured representation is often used as input to the models (e.g., Munakata, 1998; Plaut, McClelland, Seidenberg, & Patterson, 1996; Rumelhart & McClelland, 1986).

The use of structured input representations can be seen as a tradeoff between simplifying a problem to make it tractable and understanding the crucial aspects of the computations that underlie complex phenomena (see McClelland, 2009, for a discussion). However, often little effort is devoted to explaining how these representations may emerge from more basic sensory representations (but see Stoianov & Zorzi, 2012). A classic example where the type of input representations has led to theoretical controversies is Rumelhart and McClelland's (1986) model of past-tense learning. In that model, the input representation consisted of phonemes coded as wickelphones, which are the trigrams of phonemes that occur in the words. However, this input coding was strongly criticized by Pinker and Prince (1988) on a number of grounds. Another example is the Wickelgraph input coding scheme used in one of the first connectionist models of reading aloud (Seidenberg & McClelland, 1989), which was abandoned in later versions of the model (Plaut et al., 1996). Thus, defining an input coding scheme is an important and non-trivial step, and knowing more about how structured input representations are generated can help in the evaluation of a model's plausibility and descriptive adequacy.

One area where there are multiple models that use different types of representations in their input is reading aloud. They vary greatly with respect to how orthography is coded. One dimension that they vary on is whether individual letters are grouped into higher level units, such as graphemes (e.g., Perry, Ziegler, Braun, & Zorzi, 2010; Perry et al., 2007; Perry, Ziegler, & Zorzi, 2010; Plaut et al., 1996) or wickelgraphs (Seidenberg & McClelland, 1989). Graphemes, in particular, are fundamental higher level units for phonological decoding because they can be used to represent the smallest phonological units—that is, phonemes. A second dimension they vary on is how they organize the orthographic units they use. The simplest models assume that letters are used and that these are simply represented as a contiguous string (e.g., Kello, 2006). More complex representations include letters grouped into structures based on linguistic (e.g., Harm & Seidenberg, 2004; Perry, Ziegler, Braun et al., 2010; Perry et al., 2007; Perry, Ziegler, & Zorzi, 2010; Plaut et al., 1996; Zorzi, Houghton, & Butterworth, 1998a) and visual (e.g., Ans, Carbonnel, & Valdois, 1998) dimensions.

If it is assumed that orthography can be organized into higher level units and that these units can be placed into different structures, an important question arises: How is the mapping between letters and graphemes learned, and can this be learned using a relatively simple mechanism with a limited number of training exemplars? Here, a simple linear model will be developed that can learn this mapping from letter strings of any length and categorize the graphemes into onset, vowel, and coda categories, which allows these units to be assigned into a syllabically structured orthographic template. Although this model will generally be discussed in terms of how graphemic parsing could occur

and how it would integrate into the Connectionist Dual Process model (CDP; Perry, Ziegler, Braun et al., 2010; Perry et al., 2007; Perry, Ziegler, & Zorzi, 2010; Zorzi et al., 1998a; Zorzi, Houghton, & Butterworth, 1998b) and in particular the most recent version of it (CDP++; Perry, Ziegler, & Zorzi, 2010), it could be used in most models where higher level orthographic units are used (e.g., Diependaele, Ziegler, & Grainger, 2010; Plaut et al., 1996).

1.1. Levels of orthographic representations and graphemic alignment

The lowest level of input into models of reading aloud differs across models. Most models use a level whereby individual letters are represented separately from others levels (e.g., Coltheart, Rastle, Perry, Langdon, & Ziegler, 2001; Perry, Ziegler, Braun et al., 2010; Perry et al., 2007; Perry, Ziegler, & Zorzi, 2010), but others start at graphemic representations without specifying how the graphemes are computed from individual letters (Plaut et al., 1996). The CDP+ and CDP++ models (Perry et al., 2007; Perry, Ziegler, Braun et al., 2010; Perry, Ziegler, & Zorzi, 2010; Zorzi, 2010) have both a letter and a graphemic level, with graphemes, which are located in the sublexical route of the model, being activated by letters.

Evidence that graphemes and not just letters are important units comes from a number of studies, including the investigation of neuropsychological disorders (e.g., Caramazza & Miceli, 1990; Cotelli, Abutalebi, Zorzi, & Cappa, 2003; Tainturier & Rapp, 2004), experimental studies with normal readers (e.g., Rey, Ziegler, & Jacobs, 2000), and computational simulations (e.g., Houghton & Zorzi, 2003). Evidence that graphemes are likely to be activated only after individual letters, as predicted by CDP+ and CDP++, can be found in Lupker, Acham, Davis, and Perea (2012). This latter find is of importance for this study because it suggests that it is reasonable to treat letters as an atomistic level, and thus it is reasonable to assume that the activation of higher order units such as graphemes is based on letters rather than some other type of representation. Note that there are a number of effects that are generally thought to occur at the letter level, such as the effect of transposed-letters (e.g., Perea & Lupker, 2004). Simulating these effects is beyond the scope of this work. However, there are a number of models that have been specifically designed to simulate the visual front end of reading (e.g., Davis & Bowers, 2006; Gomez, Ratcliff, & Perea, 2008), and there is no principled reason for why such mechanisms could not be incorporated into more comprehensive computational reading models, such as CDP++.

Apart from the nature of the higher level orthographic units, how these units are aligned to permit an efficient mapping onto phonological units is also an important issue. This is because putting orthographic units into a simple contiguous sequence is highly inefficient as it causes dispersion in the spelling-sound correspondences (Plaut et al., 1996), and this is further exacerbated when longer multisyllabic words are used (Perry, Ziegler, & Zorzi, 2010). For example, with a word like *chalking* (/tʃɔ:kɪŋ/), if it is assumed that a parser segments the string into the graphemes ch.a.l.k.i.ng, then a simple contiguous one-grapheme-one-phoneme alignment goes off track when the –l is encoun-

tered. This occurs because the third grapheme (-l) does not commonly map onto the third phoneme (/k/) and thus all other graphemes after it also go out of alignment despite obvious relationships existing between some of them (i.e., the graphemes in –king map very commonly to /kɪŋ/). Even in a language like Italian, which has very simple spelling-sound relationships compared to English, Pagliuca and Monaghan (2010) showed computationally that using just a simple contiguous string for letter alignment leads to poorer performance than an organized template.

To circumvent the dispersion problem, some models have used more complex ways to organize orthography and phonology. One way has been to place the letters around some central point of a word (e.g., Ans et al., 1998). It is currently unclear whether such a method would allow adequate generalization performance (i.e., accurate nonword reading) in languages with difficult spelling-sound correspondences like English. Another more common method is to use an onset-vowel-coda alignment. Zorzi et al. (1998a) were the first to propose such a scheme, and it was later incorporated into many other models (e.g., Harm & Seidenberg, 2004; Perry et al., 2007; Perry, Ziegler, Braun et al., 2010; Perry, Ziegler, & Zorzi, 2010). With this type of scheme, rather than graphemes or letters being aligned based on visual characteristics of the word, graphemes are aligned based on typical linguistic categories. A major benefit of this method is that a single principle derived from spoken language (e.g., MacKay, 1971) is used to organize both orthography and phonology rather than having one method for organizing orthography and another method for organizing phonology (e.g., onset and rhymes for phonology, but a single focal point for orthography, as occurs in the model of Ans et al.).

1.2. Graphemes and alignment in multisyllabic words

If it is assumed that graphemes are aligned with lexical phonology, it means that the same graphemes do not always get placed in the same fixed order, but rather can be aligned differently depending on the phonology and the syllable structure of the corresponding spoken words. This means that the same grapheme can have different functions, depending on where it occurs in a word. In English, for example, the letter –e can function both as a vowel (e.g., *b*e*t*) and a coda grapheme (e.g., *mic*e*̲*; see also Plaut et al., 1996; who use the same distinction). Thus, if graphemes are aligned based on a syllabic structure, the letter –e needs to be put in different categories (i.e., vowels vs. codas). This is not an issue for a model like that of Kello (2006), since it does not use syllabic structure, but it is for the updated connectionist dual-process model (CDP++; Perry, Ziegler, & Zorzi, 2010), which always uses an orthographic syllable structure. With CDP++, this issue is solved by using a structure that is initially determined by basing the boundaries of orthographic syllables on lexical phonology and common grapheme-phoneme mappings. This means that with a word like *banded*, the orthographic division occurs after *ban* and before *ded* because the grapheme –n typically maps to /n/, the grapheme –d typically maps to /d/, and the syllable boundary between the two phonemes that these graphemes map into is identified from lexical phonology (i.e., /bæn.dəd/).

A problem with identifying orthographic syllable boundaries based on lexical phonology occurs when lexical phonology is not available, such as when non-words are read. In these cases, some way of approximating those boundaries must be used. One method would be to use the phonological principle of maximizing onset consonants (e.g., Hall, 2006) and apply it to graphemes rather than phonemes. For example, with the non-word *zicket*, the placement of the –ck is ambiguous, since it could be part of the first syllable (e.g., Taft, 1979) or it could be part of the second. If the maximization of onsets is applied to graphemes, then the –ck would be placed in the onset of the second syllable (i.e., *zi.cket*). This would lead to a syllabification that is the same as words of a similar orthographic structure whose orthographic representations could be based on lexical phonology (e.g., *picket, wicket*). Although onset maximization has been shown to be an effective strategy to segment orthography with CDP++, the procedure was rule based and also involved additional constraints to resolve a number of cases where simple onset maximization fails (see Perry, Ziegler, & Zorzi, 2010, for details).

The goal of this study was to investigate whether the mapping between letters and graphemes and the categorization of graphemes into useful linguistic categories could be learned using a simple connectionist network and whether this could be learned from exposure to a limited number of exemplars. In the next section of the article, we detail the model and examine its performance. In the third section, we validate the model by comparing its predictions about potentially ambiguous cases of grapheme selection against the results of a behavioral experiment. Finally, we integrate the parsing mechanism into CDP++ to assess its performance when used as the front end of a full-blown model of reading aloud.

2. A model of graphemic learning

Here, we present a new model that is able to learn the mapping between letters and graphemes and is able to categorize the graphemes such that they can be placed into an orthographic template in their correct syllabic positions. The model assumes that learning which graphemes occur in letter strings as well as their categorization can be done by one mechanism. This model is based on the idea that graphemes can fall into three different categories: onsets, vowels, and codas, and it is assumed that multiple graphemes can fall within a given category in a single word. Thus, the onsets and codas of a word may have more than a single grapheme in them (e.g., the word *shrug* contains two onset graphemes, -sh and –r). The main function of the model is to break down strings of letters of any length and number of syllables into "candidate" graphemes based on these categories. By doing this, orthographic syllables are identified, because if an onset grapheme follows a coda or vowel grapheme, it signals that a new orthographic syllable must be present. Although this mechanism makes it possible to upscale CDP++ so that it can process input representations of any length, it is worthwhile noting that any model that employs grapheme units would need a mechanism that performs this function. Thus, the present model provides such a mechanism and it could be applied to a broad class of current reading models. The model appears in Fig. 1.

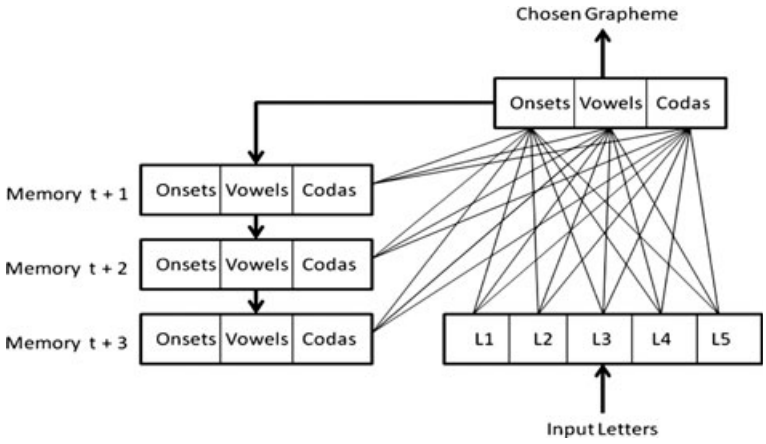

Fig. 1. The proposed model of graphemic selection and positioning. t = time; L = Letter.

There are a number of properties that the model has. One of the most important is that it is a simple learning mechanism that can potentially learn to select graphemes in any language or dialect of a language that uses an alphabet. Thus, while the model presented here is trained on the CELEX (Baayen, Piepenbrock, & van Rijn, 1993) database, which uses Received Pronunciation, nothing would stop the model learning from, for example, a database of Hoosier English.

Apart from dialect differences, since the model can learn different alignments of graphemes depending on the database, this also means that arguments about how specific phonemes are aligned are not especially important to it. For example, how some intervocalic consonants (i.e., those that occur between vowels), and in particular how the intervocalic /s/ (e.g., *respite*) should be syllabified, has generated a lot of debate in the linguistics literature (e.g., Hall, 2006; Hammond, 1999). Some authors say that it should be an onset and others a coda. With the model here, one theory could simply be selected and the /s/ presented to the model based on that theory. The model would then learn to classify the –s grapheme based on the theory and most other less contentiously aligned phonemes would be essentially unaffected. More extreme versions of what constitutes a syllable could also be used. Taft's (1979, 1992, 2001) Basic Orthographic Syllable Structure, for example, while very controversial (e.g., Perry, in press; Taft, 2001), uses orthographic syllables that are not even based on phonological representations. However, as long as the graphemes that are needed to construct the words can be determined (and Taft assumes they can be), then the model could still learn to syllabify strings of letters, albeit based on a different set of principles. The reason syllabic representations based on surface phonology are used in the model below is because that is how phonology is represented in CDP++.

A second important aspect of the model is that it needs to learn sequences of letters that correspond to graphemes one at a time starting at the beginning of a string. The use of graphemes is a simplifying assumption based on the current version of CDP++, rather than a restriction on what the model could potentially learn. There is no reason that units such

as orthographic bodies (e.g., the -ort in *fort*) could not be learned, as long as the distinction between these and onset graphemes (or larger onset units) remained. Actually investigating models that use larger units would certainly be of interest for many reasons, such as to simulate performance of people who have very poor phonological awareness at the phonemic level (Ziegler et al., 2008). This issue will be followed up in future research.

A third assumption is that the graphemes and orthographic units people learn to use are phonologically constrained. Thus, for example, the fact that graphemes are used and not simply bigrams or trigrams is because graphemes are generally associated with phonemes (Goswami & Ziegler, 2006). This has implications for the type of model used since it means that the model cannot simply learn higher order regularities from groups of letters alone and derive abstract units from these. This means that simple unsupervised learning systems that just reduce the dimensionality of data are not appropriate. Rather, lexical phonology needs to be available so that the correct sequence of graphemes can be parsed from the letter string (or at least something that specifies boundaries, such as the rules proposed by Taft, 1979). This therefore means that a supervised learning approach is reasonable to use. Note that it is also assumed that the explicit training of graphemes could help in the generation of the set of graphemes that people use, although this is not investigated here.

Given the constraints on what sort of mechanism could be used to select graphemes from a string and categorize them into onset, vowel, and coda categories, a simple linear network with a memory was used (e.g., Bartlett, Kondrak & Cherry, 2008; for a much more complex machine learning algorithm that performs syllabification). This model therefore forms a null-model against which potentially more complex variants could be tested (e.g., models that allow non-linearity). Apart from its simplicity, there are also well-known learning rules that have been thoroughly investigated that can be used to train such networks, including the delta rule which is used here (this is equivalent to the Rescorla–Wagner learning rule, see Sutton & Barto, 1981). This learning rule has been extensively investigated in terms of its properties and it has been used to model other aspects of language learning (Baayen et al., 2011; Zorzi et al., 1998a). Baayen et al. (2011) noted that it allowed their model to function as a "statistical classifier grounded in basic principles of human learning" (p. 474). It therefore represents a good starting point for modeling the mapping between letters and graphemes and, more generally, highlights the extent that the same simple principles that are embodied in the learning rule can be used to learn about and to explore a number of quite different aspects of language. More complicated models could of course be considered, such as recurrent networks that allow non-linear patterns to be learned (see e.g., Plaut, 1999, for an example of such a network being used to learn the orthography-phonology mapping), although our preference was to investigate the simplest model and learning mechanism before considering more complicated ones.

2.1. What set of graphemes should be used?

There are a number of assumptions we made when choosing the graphemes that the parser below uses. One of these is that graphemes can function in different categories

(i.e., onsets, vowels, and codas). This does not seem especially controversial, as there are clear-cut cases of this even in English (e.g., yard, by). The idea that letters or graphemes can function as both consonants and vowels in some circumstances has been incorporated not only into CDP++ but also the model of Plaut et al. (1996) and the rules of the dual-route cascaded model (DRC; Coltheart et al., 2001).

A second set of implicit assumptions is associated with the set of graphemes that were used and how they were chosen. The acquisition and learning of these was not modeled and is therefore an obvious improvement to the model. However, this is out of the scope of this article, and it represents a rather non-trivial problem. This is because what actually constitutes a grapheme and what letters should be used as graphemes has been of interest both from a modern (e.g., Venezky, 1967) and historical (e.g., Scragg, 1974) perspective. Graphemes are also likely to be learned not only from implicit information gathered when reading, but from being explicitly taught (e.g., Primary Framework for Literacy and Mathematics, 2006), and different assumptions about how they are computed has been shown to be important when used to predict experimental data from reading and spelling (e.g., Perry, Ziegler, & Coltheart, 2002; Venezky & Massaro, 1987). The set of graphemes that we used therefore represents a hypothesis about a typical set a normal reader would use.

There are four main assumptions that are incorporated into our choice of graphemes. These are as follows:

1. There are likely to be individual differences in the set of graphemes (or other units) that people use. However, we assume that our set of graphemes is very similar to the sets used by normal adult readers.
2. The graphemes people use can be based both on information that can be learned implicitly when reading and also by explicit teaching. It is assumed that explicit teaching could allow the creation of new graphemes and also prioritize some graphemes over other potential graphemes when conflicts exist.
3. All possible graphemes derivable from a database are unlikely to be used all of the time. For example, –ayor maps to a single phoneme in a very small number of words (e.g., *mayor*) and hence is highly infrequent (a very similar example is the –sw in *sword*). It is assumed here, however, that, in general, when reading non-words like *zayor,* most people would not use this grapheme because of its low frequency. This means that the grapheme is either not instantiated in its own right or that other more common graphemes would compete with it and typically win out when a choice exists.
4. While the most common graphemes are used because they commonly map to phonemes, there may be other factors that affect grapheme selection. Apart from teaching, idiosyncratic constraints may also exist, and these may be language specific. For example, there is reasonable evidence that double letters (e.g., *banner*) may be special both from a psychological perspective (e.g., Tainturier & Caramazza, 1996), and a historical language change perspective (e.g., Scragg, 1974). Thus, the extent that people may use double letter graphemes may be different than simple frequency counts or effects of teaching would predict.

Given the assumptions we have made, it is possible to speculate on the type of system that might learn such rules. First, we assume that children learn a lot about simple orthographic constraints, and thus can recognize that certain letter sequences are common and others are not (e.g., Cassar & Treiman, 1997). We also assume that most children can break words down into phonemes and hence have reasonable phonological awareness. From the two bodies of knowledge, children would develop a set of graphemes based on identifying common sequences of letters that co-occur with phonemes, and these would be constrained by orthographic knowledge they have learned. For example, with the word *chick*, there are three phonemes, and the letters that most commonly occur with them are –ch, -i, and –ck, and these would be the graphemes used to represent the word. This can be inferred because if an alternative set such as -c, -hi, and –ck was used, it would lead to a very uncommon grapheme (i.e., hi→/ɪ/) and signal that these are unlikely to be the graphemes in the word. Of course, there must be some exceptions to a pure one-grapheme-one-phoneme mapping, since there are letters sequences (notably –x) that always map to two phonemes.

Ignoring the special cases, apart from how graphemes come into existence, a question arises as to how the graphemes are assigned to words. One possibility is that they are assigned based on the most common graphemes that could be chosen based on the number of phonemes. This means that words like *chalk* would use an –al for /ɔ:/(ch.al.k) or perhaps –lk for /k/(ch.a.lk), which is unlike the graphemes that we used (ch.a.l.k) which differ in number to the number of phonemes in the word. We allowed for differing numbers of graphemes to be used because there is evidence that there can be competition between common sequences of graphemes which can occur with the same set of letters (Spinelli, Kandel, Guerassimovitch, & Ferrand, 2012). Because of this, even if graphemes come into existence based on grapheme-phoneme information, they may be chosen based at least in part from only orthographic information in cases where different alternatives can compete with each other. Thus, grapheme sequences that occur very commonly together and also consist of common graphemes that lead to a graphemic representation that has a different number of graphemes compared to the number of phonemes could potentially win out over grapheme sequences where the number of graphemes and phonemes is identical. Thus, the graphemes in *chalk* might be ch.a.l.k because –a and –l occur very commonly together and commonly map to two different phonemes, even though these phonemes are different to those found in *chalk*, and the graphemes in *sword* might be s.w.or.d rather than sw.or.d for the same reason.

One case of particular interest is the letter –e that we allowed to be used as a coda grapheme (see also Plaut et al., 1996). This grapheme does not map to a phoneme in its own right but is generally thought to influence vowel pronunciations in conjunction with another vowel (e.g., *pile*) or the pronunciations of both a consonant and a vowel (e.g., *mice*). Obviously, using a strict one-grapheme-one-phoneme alignment means that this grapheme would not exist by itself as a separate grapheme since that would, in general, cause one more grapheme than phoneme in words. Thus, if it can exist by itself as a separate coda grapheme, some hypothesis about why it can occur separately is needed.

A number of non-orthogonal possibilities could be entertained as to why the –e is separate. One is that it becomes separate because it is taught to some children that way (e.g., Lesson 5 of the reading lessons in the Primary Framework for Literacy and Mathematics, 2006). A second is because it occurs extremely commonly by itself, and thus if purely orthographic information is used when choosing graphemes, this might also allow it to be represented separately. Apart from just considering a strict dichotomy between the –e being always separate or always attached to consonants, grapheme frequency could also play a role. In this case, if frequency is important in terms of the graphemes that are used, then the –e could be represented separately in some circumstances but attached to letters to form a single grapheme in others. In the former case, if some consonant-e sequences at the end of words (e.g., r.o.be) are not represented as graphemes, then, in many cases, there would be words with more letters than phonemes (e.g., r.o.b.e -/əʊb/). One way of dealing with these sequences would be to simply assume people cannot use them for learning, like other complex sequences where it is difficult to specify the graphemes. Alternatively, a simple set of contiguous single letter graphemes could be used in learning (e.g., *r.o.b.e*) since it is easy to identify where the –b grapheme should go (it maps very commonly to/b/) and thus the alignment needed.

One final remaining question about the graphemes is actually how they are coded when learning each individual word. Our basic idea is that when children learn to read and spell, they develop a set of graphemes that can potentially be used to code words. These would initially be derived from associating common orthographic sequences with phonemes and vice versa or from being taught them (e.g., Adams, 1990). It is also assumed that, early in learning, determining these would be a largely a feed-forward processes in reading, where words are essentially read-out piecemeal style. However, later in learning, once the grapheme set becomes established, the graphemes used would be determined more automatically. This could be done via a probabilistic choice based on top-down feedback from lexical phonology, from their simple recall from links learned between the graphemic buffer and the lexical form of words, or from information stored in the lexicon (e.g., Houghton & Zorzi, 2003; Taft, 1991; Tainturier & Rapp, 2004).

2.2. The model

The model consists of a simple two-layer network (see Fig. 1) and uses identical activation dynamics as the sublexical network of CDP++. A general assumption the model makes is that, when processing letter strings, only the portion of the string that falls within an "attentional window" is available. The size of the attentional window is fixed, although it is assumed that the maximum size would differ across both languages and development (e.g., Perry, Ziegler, Braun, & Zorzi 2010), and that different reading contexts could potentially force the window to operate at less than its maximum size. The model learns to select and categorize the leftmost grapheme in this window at any given time step. A second assumption the model makes is that it stores information about graphemes that have been selected and categorized during the previous time steps (i.e., it has a memory of previous graphemes).

The output units of the network represent all possible graphemes. The multi-letter graphemes that were used are listed in Appendix S1. This set of graphemes is repeated across three slots, one for the onset position, one for the vowel position, and one for the coda position (for a total of 336 units in the output layer). The input units represent all possible letters of the alphabet plus a "null" letter used to signify that there is no letter present. This is repeated across five slots, one for each letter position of the attentional window that spans over the letter string. An attentional window of five letters was used because this is the number of letters that are needed so that the largest graphemes can be correctly selected. For example, with the –tchel sequence, determining whether the –tch grapheme should be an onset grapheme and not a coda grapheme is dependent on the last two letters (cf., *satchel* and *watched*) and thus a five-letter attentional window is needed. Apart from representing the letters, some way of representing the memory of previously selected graphemes was needed. To do this, we simply duplicated the representation in the output layer of the network and attached this to the input layer (this is similar to the context layer in the simple recurrent networks of Jordan, 1986). This was done three times so as to represent which graphemes had been selected in the three previous time steps. Three time steps were used as context since it is large enough to allow the selection of any ambiguous grapheme pattern that we are aware of. The input and output layers were fully connected.

The presentation of training exemplars was simple. The sequence for each individual letter string was as follows:

1. Up to five letters starting from the leftmost letter available were activated in the network, with one letter going into each letter slot. If there were fewer than five letters available, the remaining slots were filled by the null character. Thus, with a word like *cat*, the network would be presented *cat*** where ** represent null characters. In addition, a null character was also activated in the first memory slot (i.e., at time + 1) each time a new string was presented (i.e., with the first input pattern of the string).
2. A forward pass of the network occurred, with activation spreading to the output layer. During training, the network weights were updated based on the discrepancy between the network output and the grapheme that should have been selected. During testing, the grapheme at the output level that had the highest activation was selected.
3. The grapheme that was selected was copied into the first memory slot and those graphemes that were in the memory were updated—that is, moved back one time step, excluding the grapheme in the last memory slot, if there was one, which was simply removed from the input.
4. The attentional window was shifted forward such that the left edge of the window was placed at the first letter that was not in the previously selected grapheme (e.g., if –ch was selected in the string *chatter*, then the start of the attention window would be moved to the *a* letter, so that *atter* was then in the window).
5. The process was repeated until there were no letters left.

2.3. Training

To train the network, we selected a variable number of words from the database described in Appendix S2 as a training set (the total number of words in the database is $N = 60{,}761$). The subsets used included one using all words in the database and also a number using much smaller amounts. The smaller subsets were used to test whether the learning of graphemes in particular positions is developmentally plausible, at least to the extent that once graphemes have been learned, children are able to use them efficiently. Note that we are not claiming here that children first learn all graphemes and then start learning words as the model does. Rather, we are interested in the idea that children are not exposed to all possible words, but this does not stop them from learning to decode efficiently (Share, 1995). Therefore, the model should be able to parse strings with reasonable accuracy even when learning from a relatively limited number of exemplars. To simplify matters, we started with the entire set of graphemes we used and tested the model on the entire database. Obviously, it would be possible to use more complicated training strategies, such as simulating grapheme teaching by providing the model with high-frequency graphemes out of context. This strategy has been investigated elsewhere (see Hutzler, Ziegler, Perry, Wimmer, & Zorzi, 2004), although it is not used in this work. An even more complicated method would have been to train the model on only those graphemes that it was able to correctly parse after using an initial set to bootstrap the system and then test the model on simple sets of words children are likely to be exposed to. Additional words and graphemes could then be added iteratively to test the model at latter time points. However, this is well beyond the scope of the work presented here. Apart from this, it is also worthwhile noting that most children are likely to have learned most graphemes from quite an early age. The reading lessons used in the Primary Framework for Literacy and Mathematics previously used in the United Kingdom (2006), for example, finish teaching children graphemes by the end of Year 2 of primary school (approximately 8 years old). Therefore, presumably the orthographic forms of most words children encounter are learned after they have acquired most graphemes and thus using a full set of graphemes for all words is a reasonable simplification for our purposes.

To reduce the number of words presented to the model during training, the words in the database were first ordered by frequency. This was done to approximate a developmental trajectory whereby high-frequency words are learned before low-frequency words. From the ordered list, we selected four different subsets by using the first 500, 1,000, 2,000, and 5,000 words. These represented only a small portion of the database (0.82%, 1.64%, 3.29%, and 8.23%). Different networks were then trained for 15 cycles on each of the subsets of words, with the words being presented in their frequency order. Note that training on subsets containing only the highest frequency words in the English lexicon makes the task more difficult than training on a random subset of words. This is because high-frequency words in English tend to be shorter and have fewer syllables than low-frequency words. This means that if a full database is used for testing, then the model must not only learn from a reduced data set, but it also needs to be able to generalize the knowledge to words that are longer and have more syllables than it typically

encounters during training. With the network trained on all exemplars, the database was fully randomized for training and the network was trained for 30 cycles.

Learning was based on the simple delta rule (Widrow & Hoff, 1960), which is formally equivalent to the Rescorla–Wagner learning rule (Sutton & Barto, 1981), and the training parameters were identical to those reported in Perry, Ziegler, and Zorzi (2010) for the sublexical network of CDP++. In training, for each word, all possible sequences of letters that would fill up the attentional window and the grapheme that was the correct answer were used. Table 1 shows the exemplars created for the word *catcher* and *premise*. It is important to note that with the word *premise,* even though there is a final -e grapheme, this would not lead to the generation of a new vowel phoneme if the parser were to be integrated into a model like CDP++ (see below), since the –e is a coda and not a vowel grapheme.

2.4. Testing

The performance of the model was evaluated on each grapheme of all words in the full database at each cycle of training (418,171 exemplars), and not just the words the model was trained on. So, for example, with the word *chat*, three patterns were tested: chat* (correct answer: -ch, onset), [Memory: ch]-at***(correct answer -a, vowel), and [Memory: ch-a]-t**** correct answer: -t, coda). Performance was evaluated at the grapheme level unless otherwise stated, and when performance at the word level was evaluated, words were only considered correct if all graphemes in them were correctly generated by the model.

2.5. Results

Given its simplicity, the model showed exceptionally good performance during testing, at least in terms of error rates. The model trained on the full database had an error rate

Table 1
Exemplars created from the word *catcher* and *premise*

Word	Input Patterns	Correct Grapheme	Grapheme Categorization
catcher	catch	c	Onset
	atche	a	Vowel
	tcher	tch	Onset
	er***	er	Vowel
premise	premi	p	Onset
	remis	r	Onset
	emise	e	Vowel
	mise*	m	Onset
	ise**	i	Vowel
	se***	s	Coda
	e****	e	Coda

Note. *The null character.

of 1.27% on the full set of graphemes. In terms of words, the model made no graphemic errors (and hence got the syllabification correct) on 91.77% of the words it was trained on. This is better than the most accurate of the syllabification algorithms examined in Marchand, Adsett, and Damper (2009), and far better than those based on simple rules. For example, the top algorithm reported in Marchand et al. generated the correct syllabification for 77.68% of words, which was quite similar to our model that was trained with only 5,000 exemplars, which correctly predicted all graphemes in 74.83% of the words even though it was only trained on 31,915 exemplars (7.63% of the total). This suggests that the model not only selects graphemes accurately but also performs syllabification as well as could be expected based on the performance of algorithms reported in Marchand et al. whose *only* goal is syllabic segmentation.

It is worthwhile noting that the exceptionally good results displayed by the network here (at least in terms of error rates) using the full database for training are not trivial, as they would be if a non-linear learning mechanism that could learn essentially anything had been used. This is because the network can only learn linear relationships, and thus even though it was trained on all exemplars that it was tested on, there is no a priori reason that it should be able to reach perfect performance. What the results suggest is that most relationships between letters and the graphemes that can be formed from them are relatively simple, and thus there is no need to hypothesize that more complex mechanisms (e.g., networks with hidden units) are needed to learn them.

With regard to testing the model trained on restricted data sets, the results appear in Fig. 2. Overall, the results suggest that the model achieves reasonable generalization performance. Even when trained on relatively few exemplars (<10% of the database even when using 5,000 words, which was the largest set used), the network was reasonably accurate in selecting graphemes for many thousands of words that it never saw during learning. This was true even though English has a quite complex orthography. This suggests that, at least in terms of being able to learn from very limited data, the model is developmentally plausible in that it can learn to generalize from a small subset of data similar to that which children might be exposed to.

Apart from overall error rates, the type of errors the model makes is also of interest. The reason for this is that the type of errors displayed by the model can inform the investigation of human performance. If the model systematically makes some type of error, this leads to testable predictions about the kind of errors that people might make. If these predictions are borne out, then the learning constraints that are responsible for these errors in the model may be likely to play a role in human learning as well.

The model trained on the full database made 5,320 errors on the 418,171 exemplars (i.e., graphemes) that it was trained on. The errors were broken down into the following types[1]:

2,267 (.54%) were codas incorrectly classified as onsets (e.g., *n.i/t.w.i.t* instead of *n.i.t/w.i.t*).

1,423 (.34%) were onsets incorrectly classified as codas (e.g., *p.h.o.n/e.m.e* instead of *p.h.o/n.e.m.e* and *f.or.t.e* instead of *f.or/t.e*).

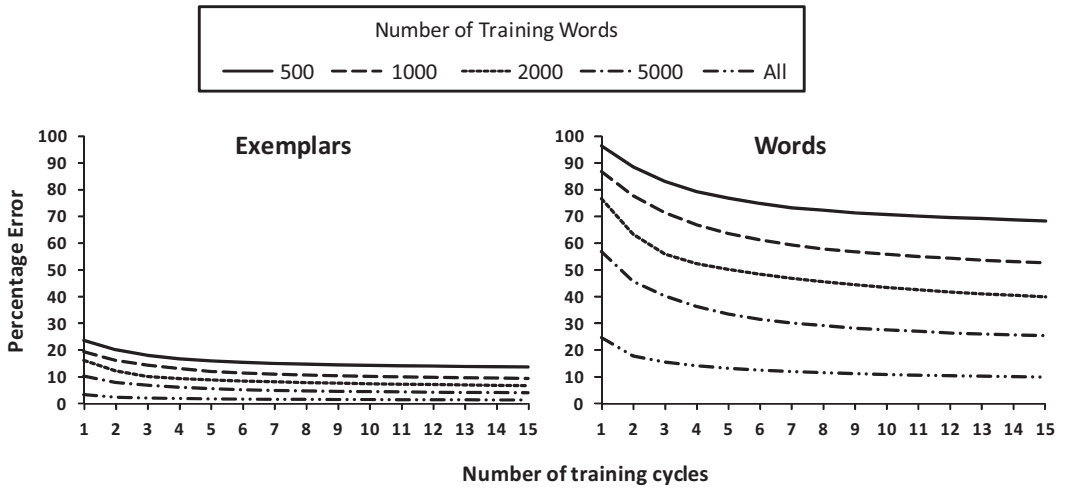

Fig. 2. Accuracy of the model on individual exemplars and words. The number is the position of the letter in the attentional window.

1,020 (.24%) were where a longer grapheme had been selected over a shorter one (e.g., *c.r.ea/t.or* instead of *c.r.e/a.t.or*).

389 (.093%) were where a shorter grapheme had been selected over a longer one (e.g., *t.o.n/g.ue* instead of *t.o.ng.ue*).

151 (.036%) were where the –e grapheme was classified as a vowel instead of a coda (e.g., *n.i/n.e/t.i.e.s* vs. *n.i.n.e/t.i.e.s*).

68 (.017%) were where the –e grapheme was classified as a coda but was in fact a vowel (e.g., *s.u.b.e/d.i/t.or* for *s.u.b/e/d.i/t.or*).

2 (.00048%) were where a different grapheme that was the same length as the correct one had been selected.

As can be seen, one of the main types of errors the model made was confusing onset graphemes for coda ones and vice versa. This type of error is to be expected for multisyllabic words that have complex intervocalic consonant clusters, since the division between intervocalic consonants is relatively complex in English (e.g., Hall, 2006). Thus, intervocalic consonants represent a more difficult aspect of defining orthographic syllables than if words with only single intervocalic consonants were examined. Some of the errors were also caused by the model failing to put a syllabic break in the correct place for polymorphemic words, instead placing the break where it typically would be found in monomorphemic words. At present, the extent that people would also show this pattern using their *sublexical* system is unclear. However, both Arciuli and Cupples (2006) and Rastle and Coltheart (2000) have suggested that it would be theoretically problematic for a sublexical mechanism to incorporate morphology. Further analysis of the performance of the model on intervocalic consonants can be found in the Supporting Information.

A second pattern of errors that is of theoretical interest was that the –e grapheme was incorrectly classified a reasonable number of times. On inspection of these words, many

created orthographic sequences that appeared orthographically plausible but were not correct based on their corresponding lexical entries. For example, the model predicted the segmentation *s.u.b.e/d.i/t.or* for *subeditor*. These results highlight two things about the model: First, that it can generate orthographic segmentations that are entirely reasonable but are not lexically correct. Second, that there are grapheme sequences in English that can be quite variable in terms of the syllabifications they occur in. It is therefore of interest to know whether this type of variability also affects the type of responses that people give or whether, for example, people simply default to a set of rule-based responses. In particular, a key prediction of the model is that graphemes such as –e can cause different numbers of syllables to be produced under different circumstances. In addition, the phonology of the different responses will be representative of two separate distributions. The model makes these predictions based on the assumption that it is integrated into a model that can generate phonology, like CDP++. In that model, graphemes cause different numbers of syllables to be produced by being put into different slots of an orthographic template. Because of this, and because different relationships are learned between graphemes and phonology in different slots, the phonology produced from sets of graphemes that are not aligned identically will differ. Practically, this is very simple to investigate compared to where syllable breaks are assigned by the sublexical route with polymorphemic words, and it is tested behaviorally in the next section.

3. Experiment

One way to test the effect of potentially ambiguous graphemes is to examine their impact on non-word pronunciation. With respect to the letter –e in English, if it is treated as a consonant grapheme, in some words, two syllables will be used (e.g., *C.l.e.v.e/l.a.n.d*), whereas if it is treated as a vowel grapheme, three syllables will be (e.g., *d.e/v.e/l.o.p*). To test this pattern, non-words with potentially ambiguous –e graphemes can be constructed (e.g., *zakemot*), and a model which classifies the -e grapheme depending on the surrounding letter context predicts that individual responses should display two types of variability: one which would be based on learning the spelling-sound mappings of words decomposed into trisyllables, and the other which would be based on learning the spelling-sound mappings of words decomposed into disyllables. Alternatively, if graphemic parsing does not create two different patterns, the responses should display variability based on what would be found if the spelling-sound mapping was learned from disyllables and trisyllables that had overlapping orthographic information. A second important aspect of this experiment is to show that people give a distribution of answers to non-words with final a–e. Finding distributions would be hard to reconcile with a rule-based mechanism of orthographic segmentation.

It is possible to understand how different predictions might be formed for different types of models by considering the predictions when using a lexicon with only two words in it, with both of them being spelt as *eve*, but where one is monosyllabic and has the same first vowel as *develop* and the second is disyllabic and has the same first vowel as *Cleveland*. If these two words are aligned identically, then one would expect the first –e

to activate the vowel phonology of both words to the same amount since there is simply nothing to distinguish between them. Alternatively, if the –ve of the two words are in different syllables, then one would expect the phonology of the first vowel to be potentially influenced by different –ve letters. This is because the –ve of the monosyllabic *eve* and the –ve of the disyllabic *eve* would not interfere with each other in learning since they are non-overlapping in terms of their graphemic position, and thus different relationships between them and the phonemes they map to can potentially be learned.

If relationships between disyllabic and trisyllabic words are learned with different grapheme sequences, then this should affect the number of long and short vowels that people produce when they give disyllabic or trisyllabic responses. This is because with trisyllabic words with single-letter first syllable vowels that are followed by a consonant and then an -e, generally the first vowel is pronounced short (e.g., *revenue;* this occurs with 86.1% of words in the current database). Alternatively, with disyllabic words with the same pattern, generally the first vowel is pronounced long (e.g., *homeless;* this occurs with 80.5% of words in the current database). Thus, a difference occurs based on the number of phonological syllables a word has (c.f., *d.e/v.e/l.o.p* and *C.l.e.v.e/l.a.n.d*). This difference is predicted because disyllabic and trisyllabic words are parsed differently. With disyllabic words, a vowel-consonant-e sequence is used in their first syllable (e.g., *h.o.m.e/l.e.ss*), and the –e generally causes single-letter vowels to be pronounced long. Alternatively, the –e acts as the vowel of the second syllable in trisyllabic words (e.g., *d.e/v.e/l.o.p*), and hence the –e does not influence the first vowel as it does in disyllabic words, and thus such a large proportion of long vowels is not expected nor found.

Apart from just disyllabic and trisyllabic words that use a vowel-consonant-e sequence, there could potentially be other types of words that affect the extent that short or long vowels are produced, such as disyllabic words without medial –e graphemes (e.g., *flippant*) and monosyllabic words with a final –e (e.g., *dome*). Thus, it is important to test the extent that people display a short-long vowel difference based on the number of syllables they produce when reading non-words that have an ambiguous –e, and this is what was done in this experiment.

3.1. Participants

Twenty-four students from a university in Melbourne participated in return for course credit. All reported having corrected or corrected to normal eyesight, all reported that they were native English speakers, and none reported having any reading problems.

3.2. Stimuli

The critical stimuli consisted of 20 non-words that had a single vowel letter in their first syllable, followed by a single intervening consonant, and then the letter –e. The consonants that followed the letter –e were all legal onset consonants. The most common single-letter vowels in English (a, e, i, o, u) were all used four times in the first syllable of the non-words. A further 195 non-words were used as fillers.[2] These were deliberately constructed so that it

would be extremely difficult to pronounce them with a different number of syllables than was planned. None used the letter –e as their middle vowel. The critical stimuli appear in Appendix S3. Nine additional filler items were used as practice items at the start of the list.

3.3. Procedure

A standard reading aloud task was used where items were presented in the middle of a screen and participants read them out aloud. Responses were recorded directly onto a second computer. Stimuli were presented using the DMDX software (Forster & Forster, 2003). Participants were told that they would see a list of non-words and that they were to read the non-words aloud as quickly and as accurately as possible. Participants were also told not to worry about errors if they made them. The non-words disappeared as soon as the microphone was triggered.

3.4. Results

Only the critical stimuli were examined. Responses where participants gave a pronunciation that was extremely divergent from those that might be predicted from the letters and hence may have been caused by other factors (e.g., the misperception of letters) were removed from the analysis as were responses that were uninterpretable for any other reason (1.46%). Individual item statistics can be found in Table S1 (Appendix S3).

The results showed that there was a large difference in the number of short and long vowels participants gave depending on whether their responses were disyllabic or trisyllabic. When participants used a disyllabic pronunciation, they most commonly used a long vowel (Long: 70.1%; Short: 29.9%). Alternatively, when they used a trisyllabic pronunciation, they most commonly used a short vowel (Long: 22.9%; Short: 77.1%). To examine this, a 2 × 2 ANOVA was used where the first factor was the number of syllables in the response (disyllabic or trisyllabic) and the second was the length of the first vowel (short or long). The absolute number of responses was used in these categories in the ANOVA and not percentage responses, which would have corrected for any differences in errors made. This was done since it was possible for participants to make no responses at all in a given cell, and hence percentage responses in those categories could not be calculated (e.g., if only trisyllabic responses were given by a participant, percentages in the disyllabic categories could not be calculated). The results showed participants preferred trisyllabic responses more than disyllabic ones (69.2%; $F1(1, 23) = 10.93$, $p < .005$; $F2(1, 19) = 17.04$, $p < .005$), and short over long vowel responses (62.2%; $F1(1, 23) = 23.40$, $p < .001$; $F2(1, 19) = 4.40$, $p = .05$). There was also an interaction between the two, $F1(1, 23) = 160.43$, $p < .001$; $F2(1, 19) = 42.54$, $p < .001$, which was due to participants preferring short vowels when giving trisyllabic responses and long vowels when giving disyllabic ones.

Overall, the results from the long/short vowel distinction are clear—whether people use long or short vowels in the first syllable of non-words is strongly affected by the

number of syllables that they use to output the non-word with. This effect was predicted based on the idea that people can organize groups of graphemes into syllables differently, and that this affects the vowel length people are likely to use in a predictable way.

4. Simulation

The experiment above showed that people give a distribution of answers to ambiguous non-words with the letter –e in them. In some cases they give disyllabic answers and in other cases trisyllabic ones. To test whether our model would produce similar results, we examined the types of segmentations it produced. To do this, all of the non-words were run through the model, and if the model selected the consonant form of the –e grapheme, it was considered a disyllabic response. If the model selected the vowel form of the –e grapheme, then it was considered a trisyllabic response. The results showed that the model produced a similar proportion of trisyllabic answers as people (Model: 65.0%; People: 69.2%).

As argued above, actually what constitutes a grapheme is not simple, and representing the letter –e separately may be controversial. An alternative way to look at this problem is whether it actually makes much difference to the performance of the model. If it does not, then whether the –e is represented separately obviously becomes a more minor issue, unless how the –e is represented can be shown to be critically important in reading and that it is only represented in one particular way across most readers. To investigate this, we therefore attached the –e grapheme to the grapheme that occurred before it in all cases in our training database and trained a new network for 15 cycles on the new representations. This meant there were no separate –e graphemes, and there were a further 41 consonant and 30 vowel graphemes that were used.

The results of the new network were first examined in terms of overall performance. The network had an error rate of 1.33%, which is very similar to the previous network. We also examined the selection of graphemes that ended with an –e. The model had an error rate of 2.51% on these, and some of these were legitimate alternative parsings. Finally, we tested the model on the ambiguous –e words used in the previous experiment. The results were identical in all ways except that the new model predicted that *badefoop* should have three syllables and not two. Altogether then, the model predicted that 70% of the responses should be trisyllabic, which is almost identical to the human data (69.2%). Thus, the results of the model appear very similar to the previous one, even though there are no single –e letters used in the coda.

4.1. Testing the graphemic parsing mechanism in CDP++: CDP++.parser

We have now described how a grapheme selection and placement mechanism could work and have examined some of the consequences of it. One way to test whether this

type of mechanism could work in a reading model is to replace that aspect of CDP++ with the model described here. This can be done simply because it is possible to assign graphemes to the graphemic buffer of CDP++ using the model proposed here, with the onset, vowel, and coda grapheme classification scheme indicating which slots in the graphemic buffer the graphemes should go. We refer this new model as CDP++.parser.

The basic idea is that instead of simply choosing graphemes from the attentional window based on the longest possible grapheme available, the parser is used to choose them instead. For example, with the non-word *chrakemot,* the attentional window will first fill up with the letters *chrak*. When this happens, a forward sweep of the network is performed, and, instead of the parser choosing –ch based on it simply being the longest grapheme (as happened with CDP++), it is instead chosen based on the –ch node in the onset category being the most activated. Based on this information, the –ch node in the first onset position of the sublexical network of CDP++ (i.e., its graphemic buffer) is activated and, after a forward sweep of that network, some phonology is produced. After a number of cycles has passed, the attentional window would have moved and then would have the letters *rakem* in it. Again, a forward sweep of the network is performed, and an onset –r grapheme is the most active. This would then be placed in the first available onset position of the sublexical network of CDP++, which is the second slot (-ch is in the first). The phonology of the sublexical network of CDP++ would then be updated via a forward sweep of the network. A few cycles later, *akemo* would be in the attention window. A forward sweep of the graphemic parser would activate the –a vowel grapheme the most. The –a slot in the vowel position of the sublexical network of CDP++ would then be activated. This process of the parser choosing and categorizing the grapheme and then putting it in the appropriate position of the graphemic buffer would then continue until the non-word had been entirely parsed.

A number of specific changes were used to integrate the new graphemic parsing mechanism into CDP++. Some of the changes were essential and some were done so that the parser would be restricted to use the maximum number of syllables in the CDP++ network (which is 2) and deal with graphemes placed in positions where the network had not learned anything (i.e., *dead nodes*) better. These were as follows:

1. Before the model starts assigning graphemes to slots in the graphemic buffer, all letters in the attentional window need to be filled or the final letter encountered.
2. Once a grapheme is chosen, the next one is chosen only once the attentional window is fully filled with letters or the final letter encountered.
3. Once the final letter of a word has been encountered, all graphemes that can be generated from those in the attentional window are selected.
4. Instead of using an onset maximization procedure, the graphemes were placed based on how they were classified according to the new parsing mechanism (i.e., in the first available onset, vowel, or coda slot of the graphemic buffer). The main exception to this was when a grapheme was placed in a "dead node" position in the onset of the second syllable—that is, a position where no spelling-sound relationship was ever learned in the network. When that happened, the grapheme

positions were revised in the same way as described in Perry, Ziegler, and Zorzi (2010). For example, with the non-word *dafvot*, the parser initially tries to assign both intervocalic consonants to the onset of the second syllable. However, because very little is learned between the –v grapheme in the second onset spot of the second syllable and phonology, the –f grapheme is moved into the coda of the first syllable and the –v grapheme moved into the first onset spot of the second syllable.

5. Because CDP++ can only deal with disyllabic words, consonant graphemes that would have otherwise occurred after the second vowel (i.e., consonant graphemes classified as onsets after a second vowel had already been placed in the graphemic buffer) were always considered coda graphemes as were –e graphemes that would otherwise have been put in a third vowel syllable.
6. We implemented an additional "dead node" strategy whereby if there was a coda –e grapheme and if consonants that followed it were placed in a dead-node position, then the coda –e was changed into a vowel –e and the consonants realigned. For example, the non-word *deseft* was initially parsed as d.e.s.e.f.t (note that it is monosyllabic). However, an –f in the third coda position cannot be processed by the network due to lack of learning. Thus, the second –e was converted to a vowel and the consonants realigned to d.e/s.e.f.t.
7. If there was a final syllable with only onset consonants in it after parsing had finished, the onset consonants were moved into coda positions of the syllable before.

To test the new model, only a restricted lexicon was used, where, for each word run, just the nodes that had an identical orthography to the word and the associated phonological word nodes that they were connected to was used. The only exception to this was for stimuli sets that require feedback and a full lexicon to be meaningful (i.e., pseudohomophone effects and simulations of phonological dyslexia), which used a full lexicon. This was done since both Perry et al. (2007) and Zorzi (2010) showed that the results of a restricted feed-forward model and a model with a full lexicon were almost identical. The model was trained in the same way as CDP++, except that all of the monosyllabic and disyllabic words that existed in the training database of the new grapheme parser were used, and the orthographic breakdown of words from that database was used also. All of the data sets that CDP++ was tested on were also tested using CDP++.parser. The results of the main databases appear in Table 2, and further simulations appear in the Supporting Information.

To evaluate the model, we first examined non-word reading performance, as documented in the Supplementary Materials. The results showed that the model did an excellent job at reading non-words. The error rates on Rastle and Coltheart's (2000), Waese and Jared's (2006), and Kelly's (2004) non-words were 5.2%, 5.0%, and 2.1%, respectively, even without excluding words that could be identified by the model as being unable to be parsed using its normal strategy because of a dead node. This suggests that learning to select the graphemes from letter strings did not impair the performance of CDP++ compared to the old method. Note that non-word pronunciations were considered erroneous when the model produced a set of phonemes that could not be generated from

Table 2
Overall r^2-values of the CDP model family on the disyllabic databases of Balota et al. (2007), Chateau and Jared (2003), and Yap and Balota (2009), and the monosyllabic databases of Balota and Spieler (1998), Seidenberg and Waters (1989), Spieler and Balota (1997), and Treiman, Mullennix, Bijeljac-Babic, and Richmond-Welty (1995)

Databases	Regression Using 8 Factors†	CDP++.parser	CDP++	CDP+	CDP
Multisyllabic					
Balota et al.	39.6	39.4	36.9		
Yap & Balota	45.7	45.9	45.4		
Chateau & Jared	38.5	40.3	33.8		
Databases	**Regression Using 3 Factors‡**	**CDP++.parser**	**CDP++**	**CDP+**	**CDP**
Monosyllabic					
SB	21.8	18.1	19.5	17.3	5.9
BS	21.5	24.3	24.0	21.6	6.7
Treiman et al.	8.2	17.4	18.1	15.9	6.5
SW	14.5	8.5	10.9	9.6	2.7

†Factors were phonetic onset coding, log word frequency, orthographic length, orthographic neighborhood, phonological neighborhood, syllable number, average syllable consistency, and basic orthographic syllable structure consistency (see Perry, Ziegler, & Zorzi, 2010, for further information).
‡Factors were log frequency, orthographic neighborhood, and orthographic length. These values were taken from Balota and Spieler (1998) and Spieler and Balota (1997).
CDP, connectionist dual process.

the graphemes based on how they are pronounced in real words. This was done because the human studies cited above did not report the individual responses and because our model is trained on a different dialect of English than that used by the participants of the three studies. Therefore, any comparison of error rates should be taken as approximate.

Apart from non-words, as can be seen from Table 2, CDP++.parser has an even higher quantitative performance than CDP++ on all of the major disyllabic databases, especially with the Chateau and Jared (2003) items. Given that the two main differences between the models are that the selection of graphemes was learned and that the graphemes were split to try and produce 1–1 relationships with phonemes in the training database, the differences, at least with the Chateau and Jared items, are likely to have been caused by that. The performance on the monosyllabic databases was generally slightly less than CDP++. However, given that, with monosyllables, the models use very similar graphemes, this difference may simply reflect slight differences in parameterization.

Finally, the performance of the model on the small experiments was also very similar to CDP++, both in terms of being able to capture the effects of interest and quantitative performance. Basically, all of the effects that CDP++ was able to capture, CDP++.parser was able to capture too. Together with the results from the databases, these results suggest that learning graphemes and the positions they go is an improvement over simply using a rule-based procedure.

5. General discussion

The way that orthography is organized is an important issue for computational models, with a number of different assumptions being made by different models. One of the main distinctions between the different models is whether orthography is fixed, where letters (e.g., Kello, 2006) or graphemes (e.g., Perry et al., 2007; Plaut et al., 1996) are always entered in the same fixed sequence, or whether higher level orthographic representations can be organized differently depending on the context of letters from which they are selected.

Here, we developed a model that learns the mapping from letters to graphemes and codes graphemes into onset, vowel, and coda categories. The model therefore not only identifies graphemes but also allows for orthographic syllable boundaries to be identified. The latter of these is achieved by assuming that a syllable break occurs whenever an onset grapheme follows a vowel or a coda grapheme. The model also assumes that the process of grapheme identification and placement occurs in two phases, one where the grapheme is identified from a letter string, and the second where it is placed into an orthographic template, which, in CDP++, is the graphemic buffer. Learning the mapping from letters to graphemes is important because it is unlikely that graphemes form the lowest level of representation that could be reasonably considered to be atomistic enough such that they could provide a reasonable starting point from which a relatively complete model of reading could be constructed. A more likely starting point would be from a letter level representation.

The results from the model showed that it was possible, with a very high accuracy, to identify graphemes and place those graphemes into their correct onset, vowel, and coda category. This was true even when only a small proportion of the full stimuli set were used in training. The ability to learn graphemes in English with a very simple model allows us to make two general conclusions. First, despite the comparative complexity of the English orthography, the orthographic regularities are sufficiently high (e.g., Adams, 1981) to exploit orthographic constraints for the computation of many useful language features. Here, we showed that it is possible to learn higher order units based only on information derived from letters. Learning graphemes in this way allows the dimensionality of this domain to be reduced as well as establishing where syllable breaks occur (e.g., Adams, 1990). Second, because we used a simple linear model (cf., Pacton, Perruchet, Fayol, & Cleeremens, 2001), the results suggest that these orthographic constraints are statistically fairly simple. Thus, it was not the case that we needed to introduce non-linear learning to reach a high level of accuracy. This also suggests that using more complex methods to learn this relationship, such as with networks that use hidden units that allow non-linear relationships to be learned, would simply be adding complexity without gaining useful insights into how the relationships between letters and graphemes are learned.

Apart from the overall performance, the model also displayed some rather interesting results in terms of the type of errors it produced. In particular, a lot of variability was produced with the –e grapheme, where it could be classified as either a vowel (e.g., *b<u>e</u>t*)

or a consonant grapheme (e.g., *mice*). With multisyllabic words, because of this ambiguity, the model would often generate orthographically reasonable sets of graphemes, but sets which would not necessarily correspond to phonemes found in the lexical forms of words (e.g., generating *r.e.v.e/n.ue* for *revenue*).

To examine whether the types of errors made by the model were similar to those people make, and thus to help validate and compare the model against other models that use simple rules to parse orthography or do not parse orthography at all, an experiment was conducted to examine people's generalization performance on non-words that are syllabically ambiguous due to an –e grapheme. The results showed that when participants gave a disyllabic pronunciation to a non-word like *zakemot*, they generally used a long vowel in the first syllable (e.g., /zeɪk.mɒt/), and when they gave a trisyllabic pronunciation, they generally used a short vowel (e.g., /zæ.kə.mɒt/). This suggests that the pronunciation of the first vowel is strongly related to the number of syllables people produce. This confirms a prediction made by the model that people parse the non-words examined into either two or three syllables (*z.a.k.e/m.o.t* and *z.a/k.e/m.o.t*), and that this is related to the phonology they are likely to use. In the first case, the vast majority of disyllabic words that use an initial syllable that uses a vowel-consonant-e pattern are pronounced with long vowels, and this is the pattern that people displayed when reading non-words. In the second case, where there is an open vowel, there is much greater variability in the pronunciations that are used in words. Models that are sensitive to the distinction between disyllabic and trisyllabic structure would therefore predict that more short vowels would be given to trisyllabic than disyllabic non-words, and that is exactly what was found.

Models using fixed representations (e.g., Kello, 2006) may have some difficulty predicting the long-short vowel pattern found in disyllabic and trisyllabic words; it is not clear how such a pattern could be produced, where, given identical inputs, words output with two syllables would typically be given long vowels in their first syllables, but words output with three syllables would typically use short vowels. At a minimum, quite complex relationships across syllables would need to be learned to determine how the vowel should be produced. In addition, with models that use a syllabically organized output (e.g., Ans et al., 1998), because two and three syllable words have different phonemes in different positions, some way of choosing the correct set would need to be done so that either a two- or a three-syllable answer could be given, rather than some blend of both.

Despite the predictions examined here, whether the difficulties noted for models that use fixed orthographic representations could be solved remains to be seen. The simplest way for determining whether the difficulties noted really are difficulties would simply be to test the models. Unfortunately, however, the model of Kello (2006) is still a prototype, and thus it is not yet fair to test the model, and the model of Ans et al. (1998) is only in French, and thus the data here cannot be simulated.

Finally, the model was tested in the context of a full reading model, CDP++ (Perry, Ziegler, & Zorzi, 2010). This was done to examine whether the new graphemic parsing mechanism, while appearing functional by itself, would still be functional when used as a component in a reading model. While CDP++ was used here, the parser could be used in other models that have an explicit grapheme level (e.g., Diependaele et al., 2010; Plaut

et al., 1996), although CDP++ is a better test of the model because it can process disyllabic words, and there is more variability in the graphemes that need to be chosen in disyllabic than monosyllabic words. The results showed that the new CDP++ (CDP++.parser), with its parser exchanged for the one developed here, had a performance level very similar to CDP++, even though an extra representational level of the model was learned rather than hard coded. These results, when combined with those of the model by itself, suggest that the rather simple mechanism proposed here provides a reasonable hypothesis for the type of computations that people might use to learn the mapping between letters and graphemes when reading.

Acknowledgments

This study was supported in part by a Swinburne Staff Development and an ARC grant (DP120100883) awarded to CP and a European Research Council grant (210922-GENMOD) awarded to MZ. The updated model as well as the Supporting Information can be downloaded at https://sites.google.com/site/conradperryshome/.

Notes

1. We use "." to represent breaks between graphemes and "/" to represent breaks between syllables.
2. A further manipulation where the list was blocked into mainly disyllabic and mainly trisyllabic halves was done to examine the effect of list context (see Perry & Jie, 2005, for a very similar manipulation). The effect of this was rather weak, and hence this factor is ignored here. This meant that two slightly different lists of fillers were used. In particular, in the two lists, 190 of the fillers were the same, with half likely to elicit disyllabic responses (e.g., *runtoid*) and half trisyllabic responses (e.g., *raspodic*). In one of the lists, an extra five disyllabic non-words were used and in the other an extra five trisyllabic non-words were used.

References

Adams, M. J. (1981). What good is orthographic redundancy? In O. J. L. Tzeng & H. Singer (Eds.), *Perception of print: Reading research in experimental psychology* (pp. 197–221). Hillsdale, NJ: Erlbaum.

Adams, M. J. (1990). *Beginning to read: Thinking and learning about print*. Cambridge, MA: MIT Press.

Anderson, J. R., Bothell, D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*, 1036–1060.

Ans, B., Carbonnel, S., & Valdois, S. (1998). A connectionist multiple-trace memory model for polysyllabic word reading. *Psychological Review*, *105*, 678–723.

Arciuli, J., & Cupples, L. (2006). The processing of lexical stress in word recognition: Typicality effects and orthographic correlates. *The Quarterly Journal of Experimental Psychology*, *59*, 920–948.

Baayen, R. H., Milin, P., Đurđević, D. F., Hendrix, P., & Marelli, M. (2011). An amorphous model for morphological processing in visual comprehension based on naive discriminative learning. *Psychological Review*, *118*, 438–481.

Baayen, R. H., Piepenbrock, R., & van Rijn, H. (1993). *The CELEX lexical database (CD-ROM): Linguistic data consortium*. Philadelphia, PA: University of Pennsylvania.

Balota, D. A., & Spieler, D. (1998). The utility of item-level analysis in model evaluation: A reply to Seidenberg and Plaut. *Psychological Science*, *9*, 238–240.

Balota, D. A., Yap, M. J., Cortese, M. J., Hutchison, K. A., Kessler, B., Loftis, B., Neely, J. H., Nelson, D. L., Simpson, G. B & Treiman, R. (2007). The English lexicon project. *Behavior Research Methods*, *39*, 445–459.

Bartlett, S., Kondrak, G., & Cherry, C. (2008). Automatic syllabification with structured SVMs for letter-to-phoneme conversion. In K. McKeown, J. D. Moore, S. Teufel, J. Allan & S. Furui (Eds), *Proceedings of ACL-08: HLT* (pp. 568–576). Columbus, OH: Association for Computational Linguistics.

Caramazza, A., & Miceli, G. (1990). The structure of graphemic representations. *Cognition*, *37*, 243–297.

Cassar, M., & Treiman, R. (1997). The beginnings of orthographic knowledge: Children's knowledge of double letters in words. *Journal of Educational Psychology*, *89*, 631–644.

Chateau, D., & Jared, D. (2003). Spelling-sound consistency effects in disyllabic word naming. *Journal of Memory and Language*, *48*, 255–280.

Coltheart, M., Rastle, K., Perry, C., Langdon, R., & Ziegler, J. C. (2001). DRC: A computational model of visual word recognition and reading aloud. *Psychological Review*, *108*, 204–256.

Cotelli, M., Abutalebi, J., Zorzi, M., & Cappa, S. F. (2003). Vowels in the buffer: A case study of acquired dysgraphia with selective vowel substitutions. *Cognitive Neuropsychology*, *20*, 99–114.

Davis, C. J., & Bowers, J. S. (2006). Contrasting five different theories of letter position coding: Evidence from orthographic similarity effects. *Journal of Experimental Psychology: Human Perception and Performance*, *32*, 535–557.

Diependaele, K., Ziegler, J. C., & Grainger, J. (2010). Fast phonology and the bimodal interactive activation mode. *European Journal of Cognitive Psychology*, *22*, 764–788.

Farah, M. J., & McClelland, J. (1991). A computational model of semantic memory impairment: Modality specificity and emergent category specificity. *Journal of Experimental Psychology: General*, *120*, 339–357.

Forster, K. I., & Forster, J. C. (2003). DMDX: A Windows display program with millisecond accuracy. *Behavior Research Methods, Instruments, & Computers*, *35*, 116–124.

Gluck, M. A., & Bower, G. H. (1988). From conditioning to category learning: An adaptive network model. *Journal of Experimental Psychology: General*, *117*, 227–247.

Gomez, P., Ratcliff, R., & Perea, M. (2008). The overlap model: A model of letter position coding. *Psychological Review*, *115*, 577–600.

Goswami, U., & Ziegler, J. C. (2006). A developmental perspective on the neural code for written words. *Trends in Cognitive Sciences*, *10*, 142–143.

Hall, T. A. (2006). English syllabification as the interaction of markedness constraints. *Studia Linguistica*, *60*, 1–33.

Hammond, M. (1999). *The phonology of English. A prosodic optimality-theoretic approach*. Oxford, England: Oxford University Press.

Harm, W. M., & Seidenberg, M. S. (2004). Computing the meanings of words in reading: Cooperative division of labor between visual and phonological processes. *Psychological Review*, *111*, 662–720.

Houghton, G., & Zorzi, M. (2003). Normal and impaired spelling in a connectionist dual-route architecture. *Cognitive Neuropsychology*, *20*, 115–162.

Hutzler, F., Ziegler, J. C., Perry, C., Wimmer, H., & Zorzi, M. (2004). Do current connectionist learning models account for reading development in different languages? *Cognition*, *91*, 273–296.

Jordan, M. I. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society* (pp. 531–546). Englewood Cliffs, NJ: Erlbaum.

Kello, C. T. (2006). Considering the junction model of lexical processing. In S. Andrews (Ed.), *From inkmarks to ideas: Current issues in lexical processing* (pp. 50–75). New York: Taylor and Francis.

Kelly, M. H. (2004). Word onset patterns and lexical stress in English. *Journal of Memory and Language*, *50*, 231–244.

Levelt, W. J. M., Roelofs, A., & Meyer, A. S. (1999). A theory of lexical access in speech production. *Behavioral and Brain Sciences*, *22*, 1–75.

Lupker, S. J., Acham, A., Davis, C. J., & Perea, M. (2012). An investigation of the role of grapheme units in word recognition. *Journal of Experimental Psychology: Human Perception and Performance*, *1*, 1–26.

MacKay, D. (1971). The structure of words and syllables: Evidence from errors in speech. *Cognitive Psychology*, *3*, 210–227.

Marchand, Y., Adsett, C. R., & Damper, R. I. (2009). Automatic syllabification in English: A comparison of different algorithms. *Language and Speech*, *52*, 1–27.

McClelland, J. (2009). The place of modelling in cognitive science. *Trends in Cognitive Sciences*, *1*, 11–38.

Munakata, Y. (1998). Infant perseveration and implications for object permanence theories: A PDP model of the A$\bar{B}$ task. *Developmental Science*, *1*(2), 161–211.

Pacton, S., Perruchet, P., Fayol, M., & Cleeremens, A. (2001). Implicit learning out of the lab: The case of orthographic regularities. *Journal of Experimental Psychology: General*, *130*, 401–426.

Pagliuca, G., & Monaghan, P. (2010). Discovering large grain sizes in a transparent orthography: Insights from a connectionist model of Italian word naming. *European Journal of Cognitive Psychology*, *22*, 813–835.

Perea, M., & Lupker, S. J. (2004). Can CANISO activate CASINO? Transposed-letter similarity effects with nonadjacent letter positions. *Journal of Memory and Language*, *51*, 231–246.

Perry, C. (in press). Graphemic parsing and the basic orthographic syllable structure. *Language and Cognitive Processes*. DOI:10.1080/01690965.2011.641386

Perry, C., & Jie, Z. (2005). Prosody and lemma selection. *Memory & Cognition*, *33*, 862–870.

Perry, C., Ziegler, J. C., Braun, M., & Zorzi, M. (2010). Rules versus statistics in reading aloud: New evidence on an old debate. *European Journal of Cognitive Psychology*, *5*, 798–812.

Perry, C., Ziegler, J. C., & Coltheart, M. (2002). How predictable is spelling? An analysis of sound-spelling contingency in English. *The Quarterly Journal of Experimental Psychology. A, Human Experimental Psychology*, *55A*, 897–915.

Perry, C., Ziegler, J. C., & Zorzi, M. (2007). Nested modeling and strong inference resting in the development of computational theories: The CDP+ model of reading aloud. *Psychological Review*, *27*, 301–333.

Perry, C., Ziegler, J. C., & Zorzi, M. (2010). Beyond single syllables: Large-scale modeling of reading aloud with the Connectionist Dual Process (CDP++) model. *Cognitive Psychology*, *61*, 106–151.

Pinker, S., & Prince, A. (1988). On language and connectionism: Analysis of a parallel distributed processing model of language acquisition. *Cognition*, *28*, 73–193.

Plaut, D. C. (1999). A connectionist approach to word reading and acquired dyslexia: Extension to sequential processing. *Cognitive Science*, *23*, 543–568.

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., & Patterson, K. (1996). Understanding normal and impaired word reading: Computational principles in quasi-regular domains. *Psychological Review*, *103*, 56–115.

Primary Framework for Literacy and Mathematics (2006). Primary Framework for literacy and mathematics. Available at: http://webarchive.nationalarchives.gov.uk/20100202100434/nationalstrategies.standards.dcsf.gov.uk/node/84445. Accessed February 17, 2013.

Rastle, K., & Coltheart, M. (2000). Lexical and nonlexical print-to-sound translation of disyllabic words and nonwords. *Journal of Memory and Language*, *42*, 342–364.

Rey, A., Ziegler, J. C., & Jacobs, A. M. (2000). Graphemes are perceptual reading units. *Cognition*, *75*, B1–B12.

Rumelhart, D. A., & McClelland, J. (1986). On learning the past tenses of English verbs. In J. L. McClelland, D. E. Rumelhart, & T. P. R. Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 2: Psychological and biological models* (pp. 216–271). Cambridge, MA: Bradford Books/MIT Press.

Scragg, D. G. (1974). *A history of English spelling*. Manchester, England: Manchester University Press.

Seidenberg, M. S., & McClelland, J. L. (1989). A distributed, developmental model of word recognition and naming. *Psychological Review*, *96*, 523–568.

Seidenberg, M. S., & Waters, G. S. (1989). Word recognition and naming: A mega study [Abstract]. Paper presented at the Bulletin of the Psychonomic Society.

Share, D. L. (1995). Phonological recoding and self-teaching: Sine qua non of reading acquisition. *Cognition*, *55*, 151–218.

Spieler, D. H., & Balota, D. (1997). Bringing computational models of word naming down to the item level. *Psychological Science*, *8*, 411–416.

Spinelli, E., Kandel, S., Guerassimovitch, H., & Ferrand, L. (2012). Graphemic cohesion effect in reading and writing complex graphemes. *Language and Cognitive Processes*, *27*, 770–791.

Stoianov, I., & Zorzi, M. (2012). Emergence of a "Visual number sense" in hierarchical generative models. *Nature Neuroscience*, *15*, 194–196.

Sutton, R. S., & Barto, A. G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review*, *88*, 135–170.

Taft, M. (1979). Lexical access via an orthographic code: The Basic Orthographic Syllable Structure (BOSS). *Journal of Verbal Learning and Verbal Behavior*, *18*, 21–39.

Taft, M. (1991). *Reading and the mental lexicon*. Hove, UK: Lawrence Erlbaum Associates.

Taft, M. (1992). The body of the BOSS: Sub-syllabic units in the lexical processing of polysyllabic words. *Journal of Experimental Psychology: Human Perception and Performance*, *18*, 1004–1014.

Taft, M. (2001). Processing of orthographic structure by adults of different reading ability. *Language and Speech*, *44*, 351–376.

Tainturier, M.-J., & Caramazza, A. (1996). Double letters in graphemic representations. *Journal of Memory and Language*, *35*, 53–73.

Tainturier, M. J., & Rapp, B. C. (2004). Complex graphemes as functional spelling units: Evidence from acquired dysgraphia. *Neurocase*, *10*, 122–131.

Treiman, R., Mullennix, J., Bijeljac-Babic, R., & Richmond-Welty, E. D. (1995). The special role of rimes in the description, use, and acquisition of English orthography. *Journal of Experimental Psychology: General*, *124*, 107–136.

Venezky, R. L. (1967). English orthography: It's graphical structure and its relation to sound. *Reading Research Quarterly*, *2*, 75–105.

Venezky, R. L., & Massaro, D. W. (1987). Orthographic structure and spelling–sound regularity in reading English words. In A. Allport, D. G. Mackay, W. Prinz, & E. Scheerer (Eds.), *Language perception and production: Relationships between listening, speaking, reading and writing* (pp. 159–181). Florida: Academic Press.

Waese, M., & Jared, D. (2006). The role of intervocalic consonants in disyllabic word naming. Paper presented at the 47th Annual Meeting of the Psychonomic Society, Houston: Texas.

Widrow, G., & Hoff, M. E. (1960). Adaptive switching circuits. In *In Institute of Radio Engineers, Western Electronic Show and Convention Record, Part 4* (pp. 96–104). New York: Institute of Radio Engineers.

Yap, M. J., & Balota, D. A. (2009). Visual word recognition of multisyllabic words. *Journal of Memory and Language*, *60*, 502–529.

Ziegler, J. C., Castel, C., Pech-Georgel, C., George, F., Alario, F.-X., & Perry, C. (2008). Developmental dyslexia and the dual route model of Reading: Simulating individual differences and subtypes. *Cognition*, *107*, 151–178.

Zorzi, M. (2010). The connectionist dual process (CDP) approach to modelling reading aloud. *European Journal of Cognitive Psychology*, *22*, 836–860.

Zorzi, M., Houghton, G., & Butterworth, B. (1998a). Two routes or one in reading aloud? A connectionist dual-process model. *Journal of Experimental Psychology: Human Perception and Performance*, *24*, 1131–1161.

Zorzi, M., Houghton, G., & Butterworth, B. (1998b). The development of spelling-sound relationships in a model of phonological reading. *Language and Cognitive Processes*, *13*, 337–371.

Supporting Information

Additional Supporting Information may be found in the online version of this article:

Appendix S1. Multi-letter graphemes used.

Appendix S2. Constructing the training databases.

Appendix S3. Percentage of trisyllabic answers produced and percentage of short vowel responses produced with two- and three-syllable responses.

Data S1. Performance of the model on intervocalic consonants.