WordKit: a Python Package for Orthographic and Phonological Featurization

Stéphan Tulkens, Dominiek Sandra, Walter Daelemans
CLiPS, University of Antwerp
Prinsstraat 13, 2000, Antwerpen
firstname.lastname@uantwerpen.be

Abstract

The modeling of psycholinguistic phenomena, such as word reading, with machine learning techniques requires the featurization of word stimuli into appropriate orthographic and phonological representations. Critically, the choice of features impacts the performance of machine learning algorithms, and can have important ramifications for the conclusions drawn from a model. As such, featurizing words with a variety of feature sets, without having to resort to using different tools is beneficial in terms of development cost. In this work, we present `wordkit`, a python package which allows users to switch between feature sets and featurizers with a uniform API, allowing for rapid prototyping. To the best of our knowledge, this is the first package which integrates a variety of orthographic and phonological featurizers in a single package. The package is fully compatible with `scikit-learn`, and hence can be integrated into a variety of machine learning pipelines. Furthermore, the package is modular and extensible, allowing for the future integration of a large variety of feature sets and featurizers. The package and documentation can be found at `github.com/stephantul/wordkit`

Keywords: Psycholinguistics, Phonology, Featurization

1. Introduction

Psycholinguistic models of word reading have been extensively investigated using computational models. Examples of such modeling efforts include the Interactive Activation (IA) model (McClelland and Rumelhart, 1981), the Triangle family of models (Seidenberg and McClelland, 1989; Harm and Seidenberg, 2004) the MROM model (Jacobs et al., 1998), the Dual Route Cascaded (DRC) model (Coltheart et al., 2001), and, more recently, the Devlex (Farkas and Li, 2002; Li and Farkaš, 2002) and the BLINCS models (Shook and Marian, 2013). All the implementations of these models, however different their core assumptions may be, have in common that they assume that words can be represented as numbers.

In spite of this reliance on feature-based representations, the debate surrounding these models has mostly focused on the way these models process their featurized word representations, not on the features themselves. The way words are featurized is usually treated as a factor which only deserves peripheral mention. This has ramifications for the way research progresses: first, it hampers empirical comparisons between models; while there is some consensus on *which* words are valid to use in simulations of psycholinguistic experiments, there is no consensus on how to featurize them. This can lead to situations in which models are deemed to be comparable because they use the same words as input, while they are actually not comparable because they featurize them in different ways (Plaut et al., 1996). Second, because features are not regarded as part of their respective models, there is a distinct lack of freely available featurizers, which hampers comparability and reproducibility.

In this work, we attempt to remedy these issues by presenting `wordkit`, a package which implements a variety of orthographic and phonological featurizers and corresponding feature sets. `wordkit` aims to be a one-stop, modular and extensible featurization package, which can be used for creating feature sets for psychological research, as well as input stimuli for computational models. The current version of `wordkit` contains three orthographic featurizers, five phonological featurizers, several orthographic and phonological feature sets, three corpus readers, and a sampler. These components are all realized as separate modules, which allows users to reorder or omit modules depending on their needs and resource availability. We will proceed as follows: In section 2. we will discuss some theoretical arguments for the use of different featurization techniques and how related work has used different featurization techniques, while in section 3. we will give a high-level overview of `wordkit`. In section 4., we will perform several experiments which evaluate the system by computing correlations between different featurization techniques.

2. Theoretical Discussion

The use of featurizers such as those included in `wordkit` touches upon at least two theoretical issues: the structure of the orthographic code, and the influence of phonology in word reading.

2.1. The orthographic code

In general, reading speeds are modulated to a large degree by the way words orthographically resemble one another; words in dense neighborhoods are, on average, read faster than words with the same length but fewer neighbors (Carreiras et al., 1997; Perea and Pollatsek, 1998; Pollatsek et al., 1999), and non-words with more neighbors are rejected more slowly than non-words with fewer neighbors (Arduino and Burani, 2004). This notion of neighborhood can be defined in several ways, but it is often defined using Levenshtein-based measures, such as Coltheart's *N* (Coltheart et al., 1977) or OLD20 (Yarkoni et al., 2008).

Regardless of how one chooses to define neighborhoods, the base assumption seems to be that words similar to the word being activated are co-activated because they have similar orthographic codes (Grainger, 2008). In this view, orthographic featurization techniques can be viewed as implementations of orthographic codes. While there is little agreement

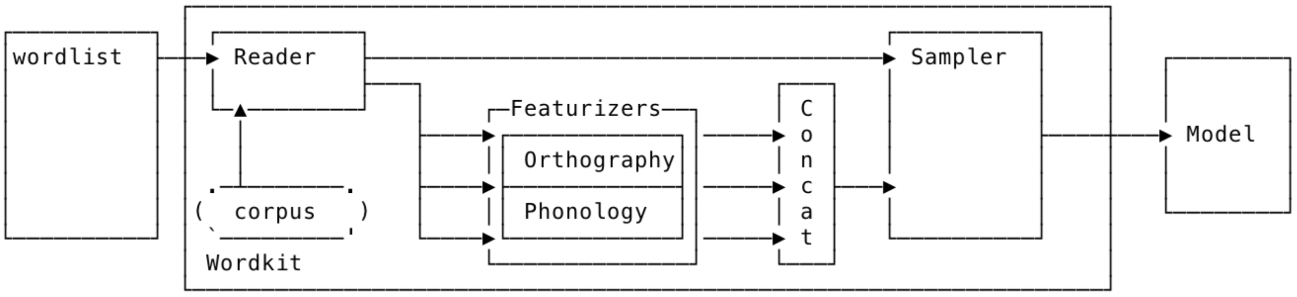

Figure 1: A sample wordkit pipeline

on which featurization technique is valid (Grainger, 2008), most models, including the Interactive Activation family of models, the DRC model and the Triangle family of models assume that the orthographic code consists of a sequence of slots, which can be filled with letters. This is striking, as a purely sequential coding of orthography is unlikely to be correct, as pointed out by Davis (Davis, 2012); readers tend to judge words with transposed or added letters as highly similar, e.g. 'garden'-'graden' and 'men'-'amen', even though the vector representations of these words are quite different when featurized using a sequential encoding.

An alternative to sequential coding is a so-called open bigram encoding (Grainger and Van Heuven, 2004; Schoonbaert and Grainger, 2004; Grainger and Whitney, 2004), which is formed by taking the ordered 2-combination of each letter in a word. That is, the open bigram encodings of the words 'salt' and 'slat' are $\{sa, sl, st, al, at, lt\}$ and $\{sl, sa, st, la, lt, at\}$, respectively. It is worth noting that the open bigram encoding has also been deemed unsatisfactory; as shown by Davis and Bowers, the open bigram coding has no way of representing whether the constituent letters of the bigrams are contiguous (Davis and Bowers, 2006). Taking the 'salt'-'slat' example above, we can see that both of the bigram sets contain the bigram at, but it is only contiguous in the case of "slat". Furthermore, Davis and Bowers have shown that participants judge substitution neighbors ('stop'-'shop') to be more similar than transposition neighbors, which is exactly the opposite of what open bigram encoding predicts (Davis and Bowers, 2006).

Another alternative to sequential coding is the spatial coding offered by the SOLAR (Davis, 2001), and the unconstrained open bigrams used by the SERIOL (Whitney, 2001) models. These featurization techniques, which offer a better fit to empirical data, are not implemented in `wordkit` because, rather than being featurizers, SERIOL and SOLAR are actually full-blown models of letter recognition. That is, SERIOL and SOLAR do not produce the type of vectors which can be used as input by other models, and are more like the models that accept the input vectors which are generated by our transformers.

In the same vein, it is worth noting that orthographic codes have also been obtained by training feed-forward supervised neural networks (Dandurand et al., 2010; Hannagan et al., 2011). While interesting from a theoretical point of view, the hidden states of such a model do not constitute the kind of representations suitable for input in most models, and, like SERIOL and SOLAR, producing these representations requires a representation of the orthographic code.

Even though strictly sequential coding and open bigram coding are unlikely to explain all observed phenomena in orthographic similarity judgments, we still include them in `wordkit`. The goal of this package is, after all, the facilitation of comparisons between different featurization techniques for replication and research purposes. A similar argument holds for the included features, which are described below. It is quite unlikely that letters are best represented by a single scalar, as in the DISLEX model of Miikkulainen (Miikkulainen, 1997), but we still include them because it facilitates reproduction and comparison between models.

2.2. The influence of phonology

While the debate surrounding the orthographic code has mostly centered on *which* information readers extract from visual stimuli, the debate in phonology has centered on the absence or presence of phonological influence during lexical access. One the one hand, this is not surprising; it is undebatable that orthographic information plays a role in reading, but phonological information is one step removed from printed words, and therefore the question of whether it influences reading is less straightforward. Furthermore, it has been extremely difficult to determine whether phonology plays a role in lexical access, as opposed to post-access; even if reliable priming effects from words to phonological neighbors are obtained, opponents of a phonological theory of word reading can still claim that these effects occur post-access, and do not influence the actual access of lexical items.

As a consequence, little attention has been given to the exact featurization of phonology in word reading. Most models that provide phonological information assume that graphemes are first converted to phonemes, after which the resulting phonemic representations co-activate similar-sounding representations (Seidenberg and McClelland, 1989; Miikkulainen, 1997; Harm and Seidenberg, 2004).

There exist two main strands in phonological featurization; one based on Onset Nucleus Coda (ONC) featurization (Keuleers and Daelemans, 2007), and the other on Consonant Vowel (CV) featurization (Li and MacWhinney, 2002). These methods differ only in the way they order phonemes, not in the way they represent individual phonemes. ONC featurizers group phonemes in subdivisions of syllables, where each syllable is sub-divided in an Onset, a Nucleus, and a

	Phonology	Syllables	Frequency
Celex	✓	✓	✓
CMU	✓		
Deri	✓		

Table 1: The corpus readers and their descriptors

Orthography	Phonology	Frequency
wind	waɪnd	298
wind	wɪnd	2170

Table 2: The structured output of 'wind' from the Celex corpus reader

Coda, while CV featurization techniques group phonemes according to a grid with alternating Consonant and Vowel slots. Note that, while these featurization techniques solve the problem of aligning phonological strings, they do not solve the problem of aligning syllabic units. That is, while phonemes within syllables are reasonably aligned when using both the CV and ONC featurizer, there is no guarantee that syllables themselves are aligned. In that sense, the problem of alignment is only partially solved by aligning phonemes to a grid Both CV and ONC transformers have been implemented in `wordkit`, and may provide a testbed for theories of phonologically driven theories of word reading.

3. Overview of `Wordkit`

`wordkit` is a modular system, consisting of discrete components, which can be combined to create an end-to-end featurization pipeline. Additionally, all these components can be used separately, or integrated into other systems. To facilitate the incorporation of modules into other machine learning experiments, all `wordkit` modules inherit `scikit-learn` (Pedregosa et al., 2011) base classes, which allows modules to be directly integrated into `scikit-learn` pipelines. Currently, there are three types of modules:

1. **Readers:** modules which retrieve words from a corpus, and output structured information about that word in that corpus.
2. **Transformers:** modules which take in structured information and turn them into feature vectors.
3. **Samplers:** modules which take a set transformed words as input and sample from this set according to their frequency.

3.1. Readers

Readers are modules which take as input strings representing the orthographic form of words, e.g. 'wood' or 'dog', and extract structured information corresponding to this string from a corpus. The information extracted from a corpus depends on the information present in that corpus: every corpus reader defines a set of so-called descriptors, which denote which data fields are contained within that corpus. We currently offer readers for three corpora: Celex (Baayen et al., 1993), a large multilingual collection of corpora (Deri and Knight, 2016), and CMUdict. The current corpus readers and their fields are listed in Table 1. If the phonological information in the corpus does not correspond to the IPA (International Phonetic Alphabet), as in the case of Celex and CMUdict corpora, we convert the representation to IPA, which serves to make these corpora comparable. As an example of use, consider the word 'wind', which is a phonologically ambiguous homograph. Table 2 shows the structured records we get as output when we present 'wind' as input to the Celex corpus reader. Corpus readers can also be combined in parallel, which allows one to, for example, perform multilingual experiments by including both the Dutch and English Celex corpora, or to combine frequency estimates from one corpus with phonological forms from another. Finally, corpus readers accept an arbitrary filtering function which filters the output given by the reader. This, for example, allows users to specify constraints on the number of syllables, frequency, word length, among others. These features make `wordkit` applicable to the creation of stimulus sets for research in experimental psychology, in which a diverse set of constraints need to be satisfied.

3.2. Transformers

In `wordkit`, a transformer is a module which turns structured information into feature vectors which can be fed into a machine learning system or otherwise used in experiments. Throughout this paper, we denote single modules as transformers, and combinations of different transformers as a featurizer. Like the corpus readers, the transformer modules can be combined in parallel, allowing for the featurization of a word into multiple feature vectors, which are then concatenated into a single vector. We will describe all transformers in order.

Linear The Lineartransformer is directly based on the featurization scheme in the Interactive Activation (IA) model (McClelland and Rumelhart, 1981), which is a slot-based featurization scheme. This implies that the orthographic code is defined as consisting of a number of slots, which are filled with feature values. In our case, the Lineartransformer represents words as concatenations of feature vectors, the values of which are determined by the feature set in use. Like other slot-based transformers, the lineartransformer makes the naive assumption that letter transpositions do not impact word recognition, and that words are strictly left-justified. This means, for example, that "stop" and "top" do not share any letters according to this encoding scheme. Nevertheless, a substantial amount of reading models, like the aforementioned IA models, assume a slot-based encoding, and its inclusion in `wordkit` is therefore warranted. The Lineartransformer can also be applied to phonological feature strings, with the caveat that this requires vowels and consonants to have the same number of features, and that vowels and consonants do not overlap in features. This is a difficult assumption to maintain, and therefore the Lineartransformer is usually not applicable to phonological feature strings.

Wickel We also include the family of Wickeltransformers, which represent words as unordered collections of *n*grams (Rumelhart and McClelland, 1985), also commonly called

	Dim
One hot	# of unique characters
Fourteen segment	14
Sixteen segment	16
Miikkulainen	1

Table 3: The available orthographic feature sets

wickelphones when used for phonology. While wickelphones have traditionally been extracted using trigrams, we extend the wickeltransformer to accept arbitrary values of n, although increasing n results in extremely large feature spaces. Wickeltransformers can be used for both phonological and orthographic strings, and do not require features to function, which makes them widely applicable. Wickeltransformers can represent order information, while still being able to flexibly account for some substitution effects. As pointed out by Rumelhart and McClelland, the Wickelphone representation can not represent one-letter substitutions correctly; the words "silt" and "slit" have 0 overlapping features (McClelland et al., 1987).
Note that using Wickelfeatures has been shown to have detrimental effects on performance in the SM89 Triangle model (Plaut et al., 1996).

Open $n$grams The open bigram coding is a featurization procedure which takes into account transposition effects by using ordered bigrams, which is equivalent to taking the 2-combination of all letters in a word (Grainger and Van Heuven, 2004; Schoonbaert and Grainger, 2004). Like in the case of the Wickeltransformer above, we extend the open bigram coding to an open $n$gram coding, in which the user can specify n herself. The Open$n$gramtransformer is also highly similar to the Wickeltransformer in other respects: it doesn't require any features, results in an unordered bag of features, and works for both arbitrary phonological and orthographic strings.

Consonant Vowel The Consonant Vowel (CV) transformer is a generalization of `patpho` (Li and MacWhinney, 2002), which represents phonology by assigning phonemes to a consonant-vowel grid, and then replacing them with binary or real-valued feature vectors. Transforming vectors using a CV grid generally applies to phonology, unless one assumes that orthographic characters are also segmented into consonants and vowels, an assumption which is rarely made[1]. The CVtransformer generalizes `patpho` because it supports variable-length grids and the use of different feature sets, while giving the same results as `patpho` when given the same grid and features. Because CV assigns words to a phoneme grid, words which do not have fully aligned orthographic forms might still be reasonably aligned in their phonological form. For example, the words 'pot' and 'spot', which share no characters in any position, are still phonologically aligned using a CVtransformer. Unlike the original version of `patpho`, our transformer can automatically optimize its grid size for a given dataset.

Onset Nucleus Coda Like the CVtransformer, the Onset Nucleus Coda (ONC) transformer creates vector representations by aligning phonemes on a grid, and replacing these by feature vectors. The main difference here is that the ONCtransformer relies on syllable information to segment phonemes into sub-syllabic Onset, Nucleus and Coda clusters. As such, this information needs to be present in the corpus if one wants to attempt featurization using an ONCtransformer; we do not provide a built-in syllabifier. Adhering to syllable structure predictably leads to a more rigid, but also a larger representation, in the sense that vectors created with the ONC transformer tend to be larger in dimensionality. Like the CV transformer, the feature grid of the ONC transformer can be automatically adapted based on the data, which means that the number of syllables, as well as the number of Onset, Nucleus and Coda clusters per syllable will be automatically determined.

[1]but see (Shook and Marian, 2013)

	Vowel Dim	Consonant Dim
patpho binary	5	7
patpho real	3	3
Plunkett	6	6
One hot phoneme	# of vowels	# of consonants
One hot feature	39	76
Miikkulainen	2	3
Binary feature	5	8

Table 4: The available phonological feature sets. Note that the One hot features and Binary features can have variable sizes, depending on the features the user decides to extract from IPA phonemes.

3.3. Feature sets

Table 3 summarizes the orthographic feature sets, i.e. the character vectors. The 14 segment display was used in the Interactive Activation model(McClelland and Rumelhart, 1981), and represents words as sets of lines, as in a rudimentary microwave display. The 16 segment display is a simple expansion of the 14 segment feature set that can also represent a wide variety of punctuation marks. The Miikkulainen feature set represents each letter by a single scalar: the proportion of black and white pixels for an uppercase version of each letter (Miikkulainen, 1997; Shook and Marian, 2013).
Table 4 details the phonological feature sets. All phonological feature sets distinguish between consonants and vowels, because the features for both types of phonemes are different. As such, allowing overlap between consonants and vowels would lead to incorrect results. The patpho features are from (Li and MacWhinney, 2002), the Miikkulainen features are from (Miikkulainen, 1997), the Plunkett features are from (Plunkett and Marchman, 1993), while the binary feature set is a modified version of the patpho feature set.
In addition, we offer the option of one-hot encoded feature sets for both phonology and orthography. The usage of these feature vectors implies that characters and phonemes are categorical, as their vectors are completely orthogonal.

3.4. Samplers

Samplers are modules which take as input structured information from a corpus which includes a frequency descriptor, as well as the featurized versions of these words. The sam-

Orth **Phon**	**Wickel** **Wickel**	**Linear (fourteen)** **CV (binary)**	**Linear (sixteen)** **CV (binary)**	**Linear (one hot)** **NGram**	**NGram** **NGram**	**Linear (one hot)** **CV (binary)**
1	logarithm	logarithm	logarithm	logarithm	logarithm	alpinism
2	rhythm	vulgarism	alpinism	allegory	allegory	aphorism
3	alga	aphorism	algerian	algeria	alligator	logarithm
4	allegory	dogmatism	aphorism	aphorism	vulgarism	vulgarism
5	gorilla	despotism	alligator	alga	glamorize	mannerism
6	amalgam	magnetism	algeria	algerian	altruism	vandalism
7	alum	anglicism	ultimatum	gallery	grim	embolism
8	al	allopathy	generator	calorie	aphorism	alga
9	algeria	mongolism	decorator	alacrity	allegoric	anarchism
10	allegoric	mesmerism	allopathy	vulgarism	lariat	galvanism

Table 5: The 10 nearest neighbors to the word “Algorithm” for a variety of featurizers. The names in braces, if any, are the feature names used by the featurizer. The feature sets and featurizers were chosen to be as diverse as possible.

pler then samples from the featurized dataset according to the frequency of these words. A sampler allows researchers to create ecologically valid datasets, insofar the frequencies in the sampled corpora are indicative of the actual frequencies with which words occur to actual language users. `wordkit` includes samplers which operate on raw frequencies, one-smoothed raw frequencies, or log-transformed frequencies.

4. Experiments

In this section we will perform two correlation experiments on a large set of featurizers. The goal of these experiments is twofold: First, we aim to show that iterating over a large set of features is easy using the provided tools in `wordkit`. Second, we aim to show that the features themselves can already account for a significant amount of variation in word reading times. The first experiment looks for a correlation between feature sets on a large (N = 17682) set of words from Celex. The second experiment uses a slightly smaller set of words (N = 10487) and correlates the average distance to differences in lexical decision times from the British Lexicon Project (BLP) (Keuleers et al., 2012). In both experiments we limit ourselves to a description of the observed differences, and leave a full-blown statistical analysis for future work.

4.1. Experiment 1

To show how the choice of features impacts the outcome of experiments, thereby demonstrating the utility of `wordkit`, we perform a correlation analysis of the distances between featurized words over a large set of featurizers. If the correlation between two feature sets is high, the choice between these feature sets makes little difference in the performance of the model. Hence, if two models use highly correlated feature sets and still show different results, the architecture, and not the features, is likely responsible for the observed differences in performance between said models. If the models use feature sets that show low correlation, any observed difference in performance using these feature sets can not be solely ascribed to architectural differences between the models.

We proceeded by first creating a set of orthographic featurizers by selecting all pairwise combinations between orthographic transformers and feature sets, which resulted in a set of six possible orthographic featurizers. Following this, we extracted all pairwise combinations between phonological featurizers and feature sets, excluding all feature sets which did not distinguish between long and short vowels (a selection of four feature sets), which led to a set of 10 possible phonological featurizers. We then defined our set of featurizers as all pairwise combinations between the set of orthographic and phonological featurizers, leading to a set of 60 total featurizers (six possible orthographic featurizers, and 10 possible phonological featurizers). Note that we did not vary any of the parameters in the featurizers, for example the values of n in the Wickeltransformer and Open$n$gramtransformer, which would have drastically increased the number of possible combinations.

We then proceeded by extracting all words whose phonological forms consisted of fewer than 12 phonemes and which were shorter than 10 characters from the English part of Celex. We also removed words if any featurizer was not able to featurize them appropriately, e.g. due to missing phonemes, special characters. This resulted in a set of 17682 words.

For each of the featurizers we calculated the cosine similarity from each word in our selected set of words to each other word, leading to a 17682×17862 square matrix for each featurizer. We then calculated the pairwise correlation between every distance matrix, leading to a set of 1770 matrix comparisons ((60 * 59) / 2). Figure 2 shows the pairwise correlations between each of the matrices. Note that the X-axis lists the names of the featurizers and feature combinations (CV, ONC, W for Wickel, and O for the Open$n$gramFeaturizer). For lack of space in the figure, we do not list the phonological feature sets, of which there were 4. These were, in order, one hot encoded phonemes, one hot encoded features, the features from Miikkulainen (Miikkulainen, 1997), and the modified set of patpho features (Li and MacWhinney, 2002).

The correlations between the distance matrices show that there exist differences between the featurization techniques: First off, the sub-orthographic features, i.e. the fourteen- and sixteen segment encodings (McClelland and Rumelhart, 1981) correlate heavily with models with the same sub-orthographic features, and erase any effect of the phonological featurizer. Also of interest is the fact that models encoded with fourteen- and sixteem segment encodings re-

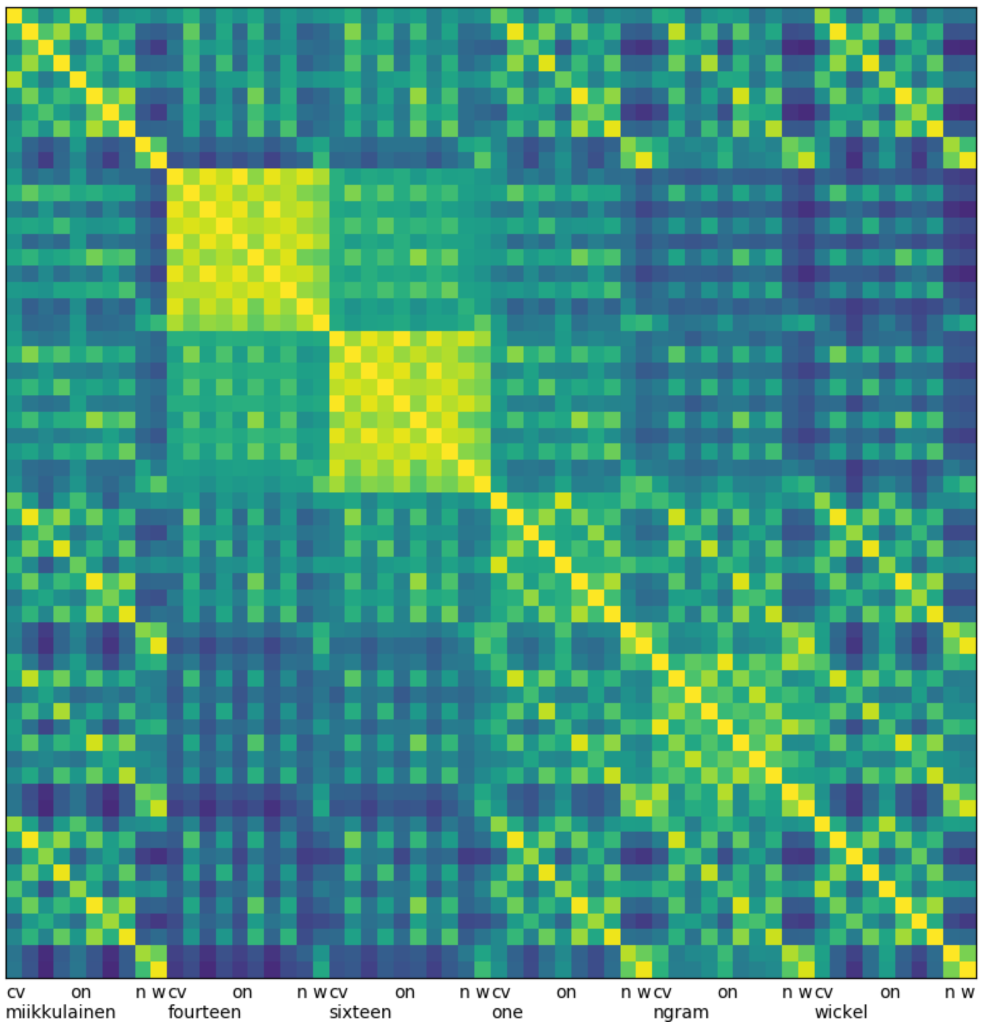

Figure 2: The correlations between featurization pipelines on a set of 17682 words, where lighter colors imply higher correlation. The phonological feature sets are not shown.

semble each other more than they do other models. This again points to a kind of primacy of sub-orthography; models with sub-orthographic features are inherently different from other models, regardless of their feature set.

A second effect is concerns phonological encodings involving ngrams, i.e. the Wickeltransformer or the Open$n$gramtransformer. Recall that these transformers do not involve any pre-defined feature sets, but instead encode stimuli using a bag of ngrams. As we can see from the figure, the correlations between featurizers which encode phonology using a Wickeltransformer or Open$n$gramtransformer are higher than the correlations between the other transformers. This again leads to a dominating effect; the effect of any orthographic encoding on the distance matrices is effectively removed, and subsumed by the effect of the phonological featurizer. In one sense this is not surprising, as these two transformers, unlike the other transformers, do not use a slot-based encoding. Looking at the effect the Open$n$gramtransformer and Wickeltransformer have on orthography, we see a less pronounced difference, although we see a larger set of correlations for the Open$n$gramtransformer than the Wickeltransformer.

Concluding this experiment, we see that both the choice of features as well as the choice of featurization techniques, greatly impacts the similarities between words, and hence the information offered to the model. More importantly, we observe that choosing a certain feature set can cause orthography or phonological similarity to dominate over other similarities. This has important implications for models of word reading; if the modeler chooses a certain feature set without validating said feature set, the modeler risks making conclusions about the relative importance of, for example, phonology, without knowing whether the features or the model architecture contributed to the observed difference.

Table 5 shows the 10 nearest neighbors to the word ”algorithm” for a variety of featurizers. Here we see that using different features and transformer combinations leads to a variety of different nearest neighbors.

Using a Consonant Vowel phonological grid in conjunction with the slot-based encoding of the linear transformer leads to a bias towards words with similar lengths. In contrast, the Wickel and NGram transformers have a lower bias towards length, and select a variety of as nearest neighbors.

Comparing the first, second, and final columns gives an idea of the influence of features, as the featurizers in each of these columns use the same features, and only differ in the selection of orthographic features. Here we see that the selection of features leads to a difference in the extracted words. This corroborates the evidence from Figure 2, in which we saw that both features and transformers influence the correlations between feature sets.

4.2. Experiment 2

In experiment 2 we used a set of 50 featurizers. We used the same featurizers as in the experiment above, except that chose to leave out the miikkulainen features on account of these not being binary features. In this experiment, we correlated the overlap distance from each word to each of its 20 closest neighbors with the Reaction Times (RT) of the BLP items (Keuleers et al., 2012). Previous work has shown that the mean Levenhstein distance to the 20 closest neighbors (OLD20) is a good predictor of reading time, as measured on multiple corpora (Yarkoni et al., 2008), and outperforms a neighborhood measure in which the orthographic neighborhood of a word is defined as the number of words within a Levenshtein distance of 1 (Coltheart et al., 1977). In this experiment, we provide a rough analogue to the OLD20 neighborhood metric, but defined on arbitrary feature spaces.

We used the same set of words as in the previous experiment, but we removed any words that do not occur in the set of BLP items, i.e. words for which we do not have reaction time measurements. This lead to a set of 10487 words. As in the previous experiment, we calculate the pairwise hamming distance between each word and each other word, leading to a $N \times N$ matrix for each featurizer.

We replaced the cosine distance in the previous experiment because the hamming distance is a straightforward analogue of perceptual distance for binary vectors. The cosine, being normalized by vector length, assigns higher weights to individual components for vectors with fewer components. This is useful when measuring the correlation between distance measures, as the magnitude of the differences between the different distance matrices no longer influences the perceived correlation.

As in OLD20 (Yarkoni et al., 2008), we define the neighborhood of a word as the mean distance to its 20 closest neighbors. Like OLD20, this neighborhood metric can then be compared to the RT measurements from the BLP. We straightforwardly chose to use the 20 closest neighbors to conform to the value reported by Yarkoni et al., although we stress that there is no intrinsic reason for choosing this value (Yarkoni et al., 2008). Additionally, Yarkoni et al. report that the choosing 20 neighbors has little effect on the explained variance of the OLD20 measure. Although we did not test this assumption, we assume that the same holds for our featurizers, and leave the investigation of this assumption for future work.

For each of the featurizers, we calculated the Pearson correlation between each word and the RT from the BLP. To get more robust statistical estimates, we bootstrapped 100 samples of 9000 words from our base set of 10487 words. Figure 3 shows the correlation and confidence intervals for each of the transformers with respect to the RTs from the BLP. For each word in each sample we also calculated the OLD20, thus also obtaining robust statistical estimates for

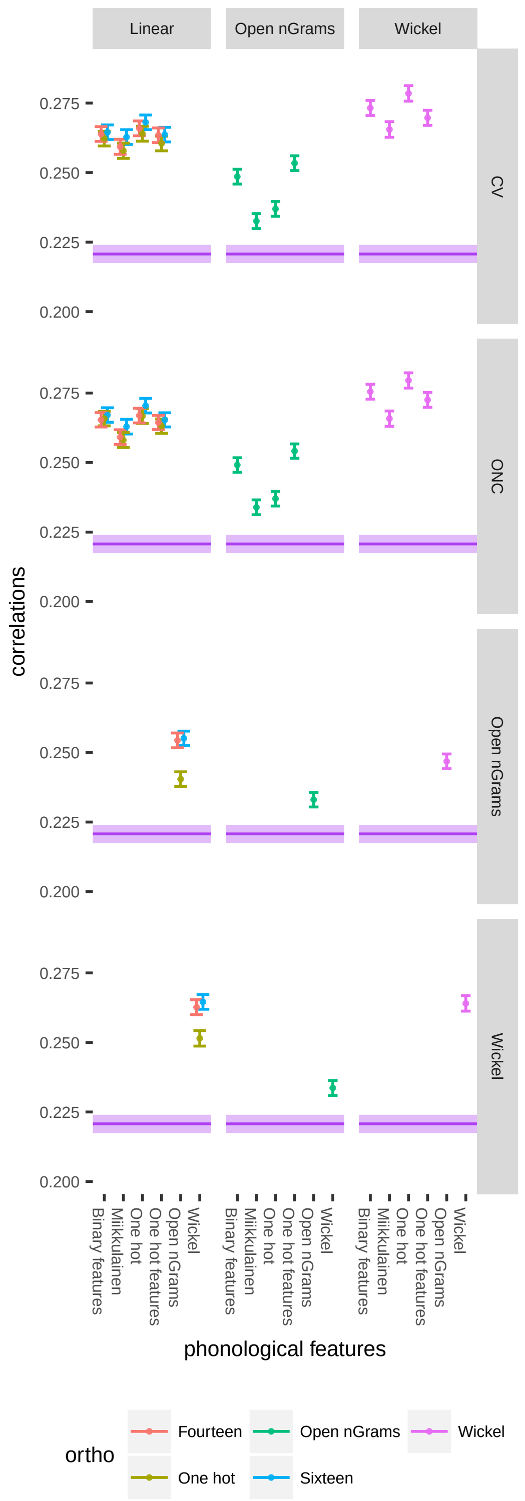

Figure 3: The mean correlations with 99.9% confidence intervals between the different transformers and the RT values from the BLP, bootstrapped from 100 samples. Each panel lists a unique Orthography × Phonological transformer combination. The columns *within* each figure specify the phonological feature sets, while the colors specify the orthographic feature sets. The purple bar at the bottom indicates the mean and confidence interval of OLD20, also bootstrapped from 100 samples.

comparing OLD20 to the other featurizers.[2]
The figure shows that all featurizers outperform OLD20. Additionally, we see that switching between the ONC and CV transformers has almost no effect on performance. It seems that simply aligning phonemes on a grid is an important factor; but that the type of grid apparently changes little in performance. Furthermore, we see that using Open nGrams as either the phonological or orthographic transformer has a severely detrimental effect on performance. Out of all transformers, OLD20 excluded, all featurizers with the lowest performance included open nGram transformers. Finally, we see that using a Wickeltransformer in the orthographic position increases the fit, and leads to the best fit overall; using it in the phonological position does not seem to increase performance. This further confirms empirical research by Davis and Bowers (Davis and Bowers, 2006): as said above, subjects tend to judge transposition neighbors to be less similar than substitution neighbors. Open bigrams, being unordered, have no way of representing whether two letters are contiguous. Similarly, the ngram encoding of the Wickeltransformer represents both order information as well as some information about some substitutions, which might explain their increased performance in this task.
One thing we did not account for in the current experiment is the presence or absence of phonological information; OLD20 is calculated only on the basis of orthographic information, while the other featurizers use a combination of orthographic and phonological information.
While this may have biased the experiment towards the featurizers, calculating OLD20 on phonological strings directly makes less sense, since phonological strings are clearly ordered in syllable clusters. Put in another way: a slot-based encoding, while unsatisfactory for orthography, is highly desirable in the case of phonology. Note that there also exists a PLD20, which is the OLD20 analogue for phonology. This measure does not work directly on phonological strings, and instead requires a metaphonetic representation.

5. Conclusion

In this work we presented `wordkit`, a versatile tool for the featurization and retrieval of word stimulus sets from a variety of corpora. We have demonstrated the utility of the toolkit by doing a large-scale analysis of different featurization pipelines, which showed that the choice of featurization pipeline leads to different nearest neighbors. As such, changing features in a model most likely has a big effect on the performance of the model. Hence, we argue that features should be seen as a part of any psycholinguistic model and the accompanying theoretical framework, and not theory-agnostic. Additionally, we performed an experiment in which we compare the performance of 60 featurizers to the well known OLD20 measure in correlating with RT judgments in a lexical decision task. This showed that there is considerable difference in performance between different pipelines, again showing that the choice of featurization is an important consideration. As for future work; we aim to expand the number of transformers and corpora in `wordkit`. We also aim to present a more thorough evaluation of the effect of orthography and phonology, and a more thorough comparison to both OLD20 and PLD20.

[2]we wrote an open-source package to estimate OLD20: `www.github.com/stephantul/old20`

6. Acknowledgments

The first author is supported by a PhD scholarship from the FWO Research Foundation - Flanders. We would like to thank Giovanni Cassani and Robert Grimm for help with the visualization and helpful discussion. Robert Grimm wrote a first version of the Python port of `patpho`, which we generalized to the CVtransformer included in the package.

7. Bibliographical References

Arduino, L. S. and Burani, C. (2004). Neighborhood effects on nonword visual processing in a language with shallow orthography. *Journal of Psycholinguistic Research*, 33(1):75–95.

Baayen, R. H., Piepenbrock, R., and van H, R. (1993). The CELEX lexical data base on CD-ROM.

Carreiras, M., Perea, M., and Grainger, J. (1997). Effects of the orthographic neighborhood in visual word recognition: Cross-task comparisons. *Journal of experimental psychology: learning, memory, and cognition*, 23(4):857.

Coltheart, M., Davelaar, E., Jonasson, T., and Besner, D. (1977). Access to the internal lexicon.

Coltheart, M., Rastle, K., Perry, C., Langdon, R., and Ziegler, J. (2001). Drc: a dual route cascaded model of visual word recognition and reading aloud. *Psychological review*, 108(1):204.

Dandurand, F., Grainger, J., and Dufau, S. (2010). Learning location-invariant orthographic representations for printed words. *Connection Science*, 22(1):25–42.

Davis, C. J. and 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(3):535.

Davis, C. J. (2001). *The self-organising lexical acquisition and recognition (SOLAR) model of visual word recognition.* Ph.D. thesis.

Davis, C. J. (2012). The orthographic similarity of printed words. *Visual Word Recognition: Models and Methods, Orthography and Phonology*, 1:120–206.

Deri, A. and Knight, K. (2016). Grapheme-to-phoneme models for (almost) any language. In *ACL (1)*.

Farkas, I. and Li, P. (2002). Devlex: A self-organizing neural network model of the development of lexicon. In *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, volume 5, pages 2546–2551. IEEE.

Grainger, J. and Van Heuven, W. J. (2004). Modeling letter position coding in printed word perception.

Grainger, J. and Whitney, C. (2004). Does the huamn mnid raed wrods as a wlohe? *Trends in cognitive sciences*, 8(2):58–59.

Grainger, J. (2008). Cracking the orthographic code: An introduction. *Language and Cognitive Processes*, 23(1):1–35.

Hannagan, T., Dandurand, F., and Grainger, J. (2011). Broken symmetries in a location-invariant word recognition network. *Neural Computation*, 23(1):251–283.

Harm, M. W. and Seidenberg, M. S. (2004). Computing the meanings of words in reading: cooperative division of labor between visual and phonological processes. *Psychological review*, 111(3):662.

Jacobs, A. M., Rey, A., Ziegler, J. C., and Grainger, J. (1998). Mrom-p: An interactive activation, multiple readout model of orthographic and phonological processes in visual word recognition.

Keuleers, E. and Daelemans, W. (2007). Memory-based learning models of inflectional morphology: A methodological case-study. *Lingue e linguaggio*, 6(2):151–174.

Keuleers, E., Lacey, P., Rastle, K., and Brysbaert, M. (2012). The british lexicon project: Lexical decision data for 28,730 monosyllabic and disyllabic english words. *Behavior research methods*, 44(1):287–304.

Li, P. and Farkaš, I. (2002). Modeling the development of lexicon with devlex: A self-organizing neural network model of lexical acquisition. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 24.

Li, P. and MacWhinney, B. (2002). Patpho: A phonological pattern generator for neural networks. *Behavior Research Methods, Instruments, & Computers*, 34(3):408–415.

McClelland, J. L. and Rumelhart, D. E. (1981). An interactive activation model of context effects in letter perception: I. an account of basic findings. *Psychological review*, 88(5):375.

McClelland, J. L., Rumelhart, D. E., Group, P. R., et al. (1987). *Parallel distributed processing*, volume 2. MIT press Cambridge, MA:.

Miikkulainen, R. (1997). Dyslexic and category-specific aphasic impairments in a self-organizing feature map model of the lexicon. *Brain and language*, 59(2):334–366.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830.

Perea, M. and Pollatsek, A. (1998). The effects of neighborhood frequency in reading and lexical decision. *Journal of Experimental Psychology: Human Perception and Performance*, 24(3):767.

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., and Patterson, K. (1996). Understanding normal and impaired word reading: computational principles in quasi-regular domains. *Psychological review*, 103(1):56.

Plunkett, K. and Marchman, V. (1993). From rote learning to system building: Acquiring verb morphology in children and connectionist nets. *Cognition*, 48(1):21–69.

Pollatsek, A., Perea, M., and Binder, K. S. (1999). The effects of" neighborhood size" in reading and lexical decision. *Journal of Experimental Psychology: Human Perception and Performance*, 25(4):1142.

Rumelhart, D. E. and McClelland, J. L. (1985). On learning the past tenses of english verbs. Technical report, la jolla institute for cognitive science.

Schoonbaert, S. and Grainger, J. (2004). Letter position coding in printed word perception: Effects of repeated and transposed letters. *Language and Cognitive Processes*, 19(3):333–367.

Seidenberg, M. S. and McClelland, J. L. (1989). A distributed, developmental model of word recognition and naming. *Psychological review*, 96(4):523.

Shook, A. and Marian, V. (2013). The bilingual language interaction network for comprehension of speech. *Bilingualism: Language and Cognition*, 16(2):304–324.

Whitney, C. (2001). How the brain encodes the order of letters in a printed word: The seriol model and selective literature review. *Psychonomic Bulletin & Review*, 8(2):221–243.

Yarkoni, T., Balota, D., and Yap, M. (2008). Moving beyond coltheart?s n: A new measure of orthographic similarity. *Psychonomic Bulletin & Review*, 15(5):971–979.