Text to Phoneme Alignment and Mapping for Speech Technology: A Neural Networks Approach

John A. Bullinaria

Abstract— **A common problem in speech technology is the alignment of representations of text and phonemes, and the learning of a mapping between them that generalizes well to unseen inputs. The state-of-the-art technology appears to be symbolic rule-based systems, which is surprising given the number of neural network systems for text to phoneme mapping that have been developed over the years. This paper explores why that may be the case, and demonstrates that it is possible for neural networks to simultaneously perform text to phoneme alignment and mapping with performance levels at least comparable to the best existing systems.**

I. Introduction

Many speech technology applications rely on having a good mapping from text to phonemes that not only performs perfectly on known words, but also generalizes well to new words, such as previously unseen proper nouns [8].

Alignment of the text and phonemes is the first stage of data processing necessary to provide useable training data for many text to phoneme conversion systems, including the most successful symbolic rule-based systems [8, 10] and most neural network systems [14, 16, 17]. The state-of-the-art for this alignment process appears to be the rule-based Expectation-Maximization (EM) algorithm of Damper et al. [9], and data aligned in that way has been employed in the rule-based Pronunciation by Analogy (PbA) system of Damper et al. [8, 9] to provide state-of-the-art text to phoneme mappings for English [10].

Given the number of neural network based systems that have been developed over the years, it seems surprising that they are not more competitive in this area. One reason is that most of the older neural network models [3, 6, 14, 16] were aimed mainly at modeling psychological data and understanding human language abilities, rather than producing high performance applications for speech technology. They therefore concentrated on producing human-like performance on small-scale empirically testable data-sets rather than large-scale systems useable for real world applications. Moreover, the computational resources required for training such neural networks has prevented scaling them up to larger data-sets. However, computers are now much more powerful, and a simple scalable neural network based approach for dealing with both the alignment and mapping problems has existed in the psychological modelling literature for some time [3, 4, 5, 6], so it is worth exploring what can now be achieved with a neural network approach to this problem.

John A. Bullinaria is with the School of Computer Science, University of Birmingham, Edgbaston, Birmingham, West Midlands, B15 2TT, UK. (e-mail: j.a.bullinaria@cs.bham.ac.uk).

In the remainder of this paper, it is described how that neural network approach works, various issues are addressed that relate to scaling up from simple psychological models to practical large scale alignment and mapping systems, and the results achieved are compared with those obtained with the Damper et al. approach [9]. The advantage of the approach here is not only in the good performance, but also in the fact that the text to phoneme conversion neural network performs the alignment as part of the standard learning phase.

II. The Basic Neural Network

Whether building psychological models of reading, or more practical text-to-phoneme conversion applications, aligning the text (i.e. letters/graphemes) and phonemes is a crucial first step. As Damper et al. [9] discussed in detail, some individual letters (e.g., “X”) can sometimes correspond to more than one phoneme, and other strings of letters can correspond to only one phoneme (e.g., “TH”). Consequently, given a string of letters and the corresponding string of phonemes, it is not always obvious which letters map to which phonemes. Early neural network systems such as the NETtalk model of Sejnowski & Rosenberg [17] used pre-aligned training data to bypass the problem. Later models by Seidenberg & McClelland [16] avoided pre-alignment by being based on Wickelfeatures (letter and phoneme trigrams) but had relatively poor performance levels. Then Plaut et al. [14] developed improved models, but returned to pre-aligned training data to get good performance. This led Bullinaria to explore extensions of the NETtalk model that were able to learn appropriate alignments at the same time as learning the mapping process [3, 5, 6].

Those models used a standard fully connected feed-forward neural network architecture [2] similar to the original NETtalk model [17], with one block of $N_{letters}$ input units (each representing one letter of the alphabet) for each potential input letter, one block of $N_{phonemes}$ output units (each representing one phoneme type) for each phoneme output, and N_{hidden} hidden units in between. Each input word starts off with its first letter aligned with the middle input block, and then slides to the left, one letter at a time, until its last letter is aligned with that block. The network is trained so that, at each stage, the output phonemes correspond to the letter represented by the central input block, in the context of the letters surrounding it. Letters corresponding to more than one phoneme are accommodated by having more than one output block, and strings of letters corresponding to shorter strings of phonemes are accommodated by allowing a “null phoneme” or “blank output” to be represented along with the real phonemes. For standard English words, two blocks of

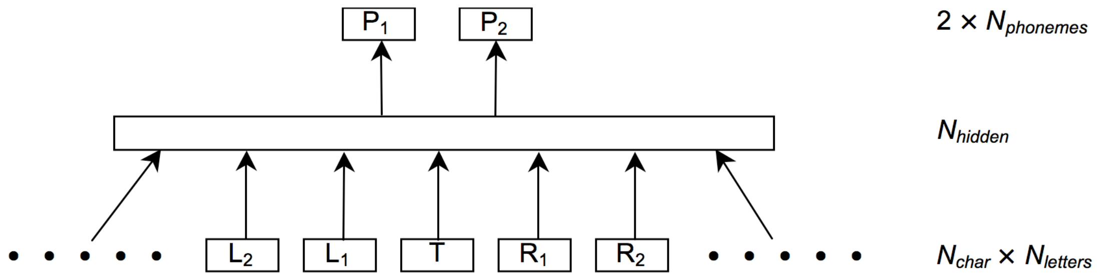


Fig. 1. The neural network architecture with two output phonemes per word presentation. The output phonemes P_1 and P_2 correspond to the target letter T in the centre of the input window, in the context of the letters L_1, L_2, … to its left and the letters R_1, R_2, … to its right.

phonemes are sufficient, but this will result in a few words (such as "XMAS") having less than optimal alignment. These considerations lead to the neural network architecture shown in Figure 1.

For pre-aligned training data one can use a standard neural network learning algorithm [2] such as traditional gradient descent connection weight updating (the back-propagation algorithm) so that the correct outputs are produced for each word in each input position. A more sophisticated learning algorithm is clearly required if it must do the alignment too. As Damper et al. [9] explained in detail, there is no unique correct alignment (e.g., the phoneme corresponding to a "TH" could equally well be aligned with the "T" as with the "H"), but some alignments are clearly more appropriate than others. Consider the word "THAT" which maps into three phonemes: "dh ae t" in the BEEP notation [15]. If "_" represents a blank (null phoneme), then either "dh _ ae t" or "_ dh ae t" would be equally acceptable alignments, but "dh ae t _" would not be, because mapping the letter "A" to the phoneme "t" would be highly irregular compared with mapping the letter "T" to the phoneme "t". The key idea here is the concept of *regularity*: good alignments are highly regular, whereas poor alignment contain many irregularities. Fortunately, neural networks are very good at identifying regularities in data, and using them to their advantage.

This led to the idea of *multi-target training* [3, 5] in which each input has a whole set of possible output targets, but the network is trained only on the target that already exhibits the lowest output error (e.g., the lowest sum-squared difference between the actual network outputs and the target outputs). If the multiple targets are the full set of possible text-to-phoneme alignments, then, even starting from random initial network weights, this process can settle down to using the set of targets that correspond to a good regular set of alignments. The reason this works is that, even with totally random starting alignments, the coherent regular weight updates tend to build up, while the irregular incoherent weight updates tend to cancel out, with the net effect that the regular alignments will have their output errors reduced by more that the irregular alignments. This, of course, means that in the next round (epoch) of updates, the more regular alignments will tend to have lower errors and be chosen as targets, rather than the irregular alignments, and the good alignments will dominate the weight updates even more. This process snowballs until the good alignments totally dominate the chosen targets, and the alignment process is complete. The same good set of alignments are then chosen every time until all the output activation errors are reduced to their chosen low levels.

This basic architecture and learning algorithm formed the basis of a series of models of reading aloud, including human-like generalization ability for reading non-words, accounts of frequency and regularity effects in reaction times, and models of developmental and acquired surface dyslexia [3, 6]. Relatively straightforward extensions of it were also used to model the harder reverse task of spelling, i.e. phoneme-to-text conversion [4, 6]. One might wonder, therefore, why this process has not been applied to text-to-phoneme mapping more generally. The reason is probably that those earlier models were all rather small scale, and there are various practical technical issues that need to be addressed before it can be applied to large scale datasets such as the British English Example Pronunciation (BEEP) dictionary [15] used in the Damper et al. [9] study.

III. Scaling Up the Neural Network

The main scaling-up problem one faces is the combinatorial explosion in the number of possible target alignments for longer words, which proves particularly troublesome for words that are more than eight or nine letters long. In the BEEP dictionary [15] there are approximately 200,000 words, up to 28 letters long, with over 25% of them eleven or more letters long. This means that, in practice, one will eventually need to restrict the number of possible alignments that are considered, but to do that in such a way that it catches a large proportion of the best alignments.

Another problem one has with very large training data sets is that the network weight updates corresponding to highly irregular mappings (e.g., the letters "LB" being pronounced "p aw n d") are swamped by those coming from more regular mappings, and this renders them extremely difficult to learn. In psychological models this is usually less of a problem, because one takes word usage frequency into account in the training process, and since irregular words tend to be higher frequency as a result of language evolution [12], this tends to compensate sufficiently [6, 14, 16]. When word frequency information is not available (e.g., as with the BEEP dictionary data), an alternative approach is to stop updating the weights for words that have already reached some threshold output error level (e.g., that have each output unit activation within 0.2 of its target value). This proved to work very well and was adopted as standard here.

There are also a number of very long range dependencies that have to be dealt with. For example, in the words

PRECONSIDERATIONS	⇒	p r iy k ax n s ih d ax r ey sh n z
RECONSIDERATIONS	⇒	r iy k ax n s ih d ax r ey sh ax n z

the pronunciation of the "ION" depends on the initial "P". BEEP is generated from many different sources, and is known to contain errors [13], so it is debatable whether this is a real idiosyncrasy of English, or simply an error in the dictionary. Either way, the algorithm will need to deal with it and have a big enough input window to accommodate the "P" while the "ION" is in the centre position. To cope with this, an input window of 41 letters was used, with 20 context letters each side of the crucial central position. This results in a total of 1066 input units, and quite large neural networks to train. This could cause computational resource problems for a naïve implementation of the network, but in practice it is possible to simply ignore at each stage any input units that are not activated, because they do not contribute to any activations or weight updates. This means using, on average, only nine inputs rather than the full 1066.

A side effect of needing a large amount of context information for some words, is that it may be misused for other words. Underlying the concept of regularity here is the idea that the text-to-phoneme mapping consists of a whole hierarchy of rules embodied in the neural network connection weights, and that the simpler and more general that rule set is, the better it is likely to generalize. This implies making the *minimum* use of the available context information that is consistent with performing the mapping accurately. In other words, long range context information should only be used when more local context is insufficient. Since the gradient descent learning in neural networks will update the weights connected to any useable input information, this could be problematic. One solution would be to have different learning rates for the weights connected to different blocks of inputs: large for the central block, and increasingly smaller for more distant blocks. This does work, but it proves extremely difficult to have the rates fall off with distance sufficiently fast without considerably slowing down the training time of the network. A simple alternative solution derives from the idea that children are generally taught to read shorter words before being exposed to longer words. This actually works extremely well for training the neural networks. The system starts off learning words of length one, and only when a fixed proportion (95% in the standard case) have been learned correctly, does it move on to words of length two or fewer. At each stage it waits till 95% are learned correctly, before increasing the maximum word length by one more. This continues until it is training on all the words, and then it keeps going until all the words have been learned.

This incremental learning approach also proves useful for dealing with the combinatorial explosion of the target alignments mentioned above. One can easily deal with the different word length sets in different ways. There are only 829 words of three or fewer letters, and these are easy to align by hand. The earlier studies [3, 5, 6] showed that this was not actually necessary for achieving good alignments, but it does allow the removal of some of the arbitrariness of the resulting alignments, which makes the subsequent analysis and evaluation easier. In particular, for the hand-aligned words a non-blank is only allowed in the right-hand phoneme block when there is already one in the left-hand block, and when two letters map to a single phoneme, that phoneme is always aligned with the left most letter. Of course, once the networks move on to the longer unaligned word-sets, which no longer include fixed alignments for the words of three or fewer letters, it will be free to change the alignments it was forced to learn previously, but it will only do that if there is some advantage in it.

As the network moves on to learning each new larger word-set, most of the shorter words are already pronounced correctly, and many of the new words are also already correct without needing any further training, because they are regular and the neural network automatically generalizes well to them. There is clearly no need to check for better target alignments in any of those cases. This means that for word-sets with a maximum of up to 15 letter words, it is actually computationally feasible to search through all the possible alignments for each incorrect word at each stage of training. For longer words, a full search does become impractical. However, by then, around 195,000 words (i.e. about 98% of the whole training set) will have already been aligned and learned correctly, and the principal alignment rules are firmly established in the network weights. Many other words will only be pronounced incorrectly because of subtle variations in the vowel sounds, but their alignment will be correct. Beyond 15 letter words, there are actually no new words introduced that have no letters at all aligned and pronounced correctly, and those letters that are correct will strongly restrict how the rest are aligned. It turns out that all one needs to do is identify the existing non-blank outputs for each word, and "patch-up" the errors to result in the right number of output phoneme positions, and those provide a very good indication of where each target phoneme should go. Then going though a simple check of how swapping each blank with each non-blank affects the total error, gives a confirmation that the alignment is good, and occasionally one can find an improved alignment to use. This faster process still sounds computationally costly, but since all the output activations are already computed, it usually has negligible cost compared with passing the activations through the network and computing the outputs in the first place. Even if this mechanism does not identify the best possible alignment for a particular word, it will not be totally wrong, and the combination of weight updates for all words will improve the alignments ready for the next round. In the same way that random initial alignments sort themselves out, so do any less than perfect choices of output targets here.

IV. Implementational Details

Neural networks are usually started with all their weights and biases drawn from the same uniform distribution of random numbers. However, it is now known that more complicated set-ups tend to emerge in studies in which different initial weight distributions for each part of the network are optimized by processes such as simulated

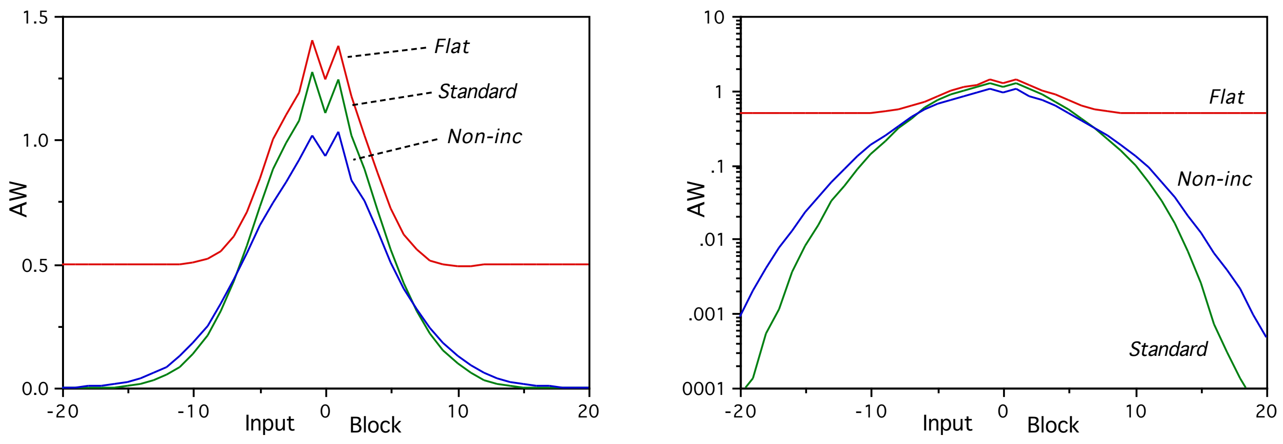


Fig. 2. Average absolute weight magnitudes (AW) for the 41 input blocks, on a linear scale (left) and log scale (right). The weights for the long range context blocks are lowest for the standard network, more for non-incremental learning, and highest for a flat initial weight distribution.

evolution by natural selection [7], so it is worth considering more carefully what is most appropriate here. As noted above, when one thinks in terms of the hierarchy of rules determining the correct phoneme outputs for each letter, it is reasonable to expect that it would be best to only use long range context information when the more local context is insufficient. In neural network terms, that would imply that the average magnitudes of the weights connecting the more central input blocks to the hidden units should be higher than those connecting the less central input blocks. Ultimately, it is the learning algorithm which determines the weight values, but it may help if the random initial values for the input to hidden unit weights are not set too high as a starting point. This is particularly important for the incremental approach adopted here, to prevent any disruption caused by the increased number of context blocks suddenly introducing additional random activations into the network each time the maximum word length is increased. The standard neural network approach here takes this idea to the extreme and has all the initial input to hidden layer weights start from zero, to minimize any unnecessary contributions from the outer context blocks. It also starts all the hidden unit and output unit biases from zero. Only the hidden unit to output weights start from a traditional uniform distribution of random values in the range [-1,+1]. This will later be compared empirically with the more conventional approach of starting all the network weights and biases with a flat distribution of random values in the range [-1,+1].

Given the above architecture and learning algorithm, there remain a few other details that need to be specified. First, one must set an appropriate number of hidden units for the neural network. Obviously, there needs to be enough units that the associated connection weights can accommodate all the pronunciation rules, but not so many that the network takes too long to train. One does not have to worry too much here about restricting the numbers to avoid over-fitting – in this case the "noise" in the training data is the irregular pronunciations, and we do want to learn them. A good compromise for the BEEP data was found to be 2000 hidden units. The earlier studies [3, 5, 6] all used the standard sum-squared error measure for the gradient descent learning, but it has since been established that the cross-entropy error measure works better for binary mappings [2, 7], so that is used here. Finally, a gradient descent learning rate of 0.1 proved to result in learning that was reasonably fast, but not so fast that networks settled prematurely on inappropriate alignments, or kept on switching unnecessarily between different target alignments.

V. Neural Network Learning Results

The neural network system described above was trained on 198618 words from the BEEP dictionary [15], consisting of the same 198632 words used in the Damper et al. study [9] except for 14 very non-standard "words" that had many more phonemes than letters (e.g., "X" pronounced "e k s", and "UNIV" pronounced "y uw n ih v er s ih t iy") which was a clear indication that they deviated considerably from standard English pronunciation. Twenty such runs, each using different random number seeds, and numerous further sets of runs designed to explore variations in the details and parameters specified above, all achieved 100% correct pronunciation on the training data.

The first issues explored were the effects on the final learned weights of the non-traditional distribution of initial weights and the incremental learning approach. Figure 2 shows the final average absolute weight distributions for the standard networks, the equivalent networks that start from a more conventional flat (uniform) distribution of initial weights, and the equivalent networks that learn from all the training words throughout, rather than the incremental process that has the shorter words largely learned before moving on to longer words. It is clear that the standard network does, as hoped, make less unnecessary use of the longer range context information.

Of course, it is not the weight distributions that are most important, but the quality of the resulting network outputs. A crucial question for this paper is: how well do the resulting alignments compare with the Damper et al. approach [9]? In their paper, they compared the various alignments by measuring the corresponding generalization performance on

	C Average	*C* Std. Dev.	*C* Maximum	Ensemble *C*
Standard NN	0.5606	0.0012	0.5619	0.5630
Non-incremental NN	0.5553	0.0055	0.5625	0.5560
Flat initial weights NN	0.5590	0.0013	0.5613	0.5610
MT only to length 7 NN	0.5610	0.0014	0.5620	0.5629
No MT at all NN	0.4868	0.0076	0.5012	0.4934
Naïve Alignment	0.2276	–	0.2276	–
Damper et al. [9]	0.5630	–	0.5630	–

Table 1. Alignment consistencies *C* (average, standard deviation, maximum and ensemble average over 20 runs) for the standard neural network studied in this paper, four variations of it, the naïve alignment, and the alignment generated by Damper et al. [9].

the Pronunciation by Analogy (PbA) system [8]. It is not totally obvious that such pronunciation performance is really the best measure of alignment. All the neural networks here achieved 100% on the training data, despite spanning a range of different qualities of alignments, but it is reasonable to expect that a more consistent or regular alignment will allow a better hierarchy of rules to emerge (particularly in a rule based system like PbA) and that will result in better generalization. Such a measure will therefore be considered in the next section, but first it seems prudent to consider a more direct measure of alignment quality.

Actually, Wolff, Eichner & Hoffmann [18] have already presented a suitable grapheme to phoneme consistency measure. The idea is that, given aligned strings of graphemes $g \in G$ and phonemes $f \in F$, and associated probabilities $p(g)$, $p(f)$ and $p(g,f)$, one can define the entropy

$$H = -\sum_{g \in G, f \in F} p(g,f) \log p(g,f)$$

and mutual information

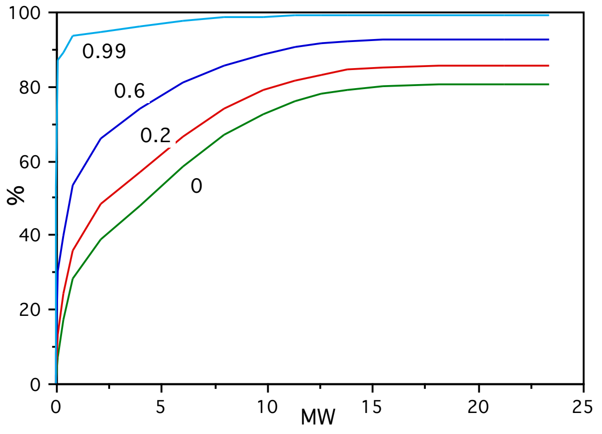
$$I = \sum_{g \in G} \sum_{f \in F} p(g,f) \log\left(\frac{p(g,f)}{p(g)p(f)}\right)$$

so that $C = I/H$ is a measure of mapping consistency.

If the *G-F* mapping were totally random, that consistency *C* would be zero, while a perfect 1-1 mapping with each grapheme corresponding to a single distinct phoneme would have a consistency of one. Natural languages fall somewhere in between. Wolff et al. [18] suggest that German and Dutch have consistencies around 0.75, and English around 0.65. However, large dictionaries containing mixtures of regional variations and a significant number of errors will have lower consistencies, even if the alignment is carried out as well as possible. Moreover, the sizes of the grapheme and phoneme sets can vary across different dictionaries or representations for the same language [9]. This renders absolute consistency evaluation difficult, but *C* can certainly be used to compare different alignments on the same dictionary.

The neural networks map individual letters to either zero, one or two phonemes. It therefore makes sense to take the "graphemes" *G* here to be the letters, and the "phonemes" *F* to be the set of possible outputs. Table 1 summarizes the alignment consistencies *C* then achieved by the key neural networks studied, showing the averages, standard deviations, and maxima over 20 runs. The standard neural network here achieves a consistency of 0.5606, which is significantly better than both the non-incremental learning approach (0.5553, unpaired two-tailed t-test, $p = 0.0004$) and the flat initial weight distribution approach (0.5590, $p = 0.0005$). A computationally faster variation that switches from the full multi-target learning approach to the simplified "patch-up mechanism" at word length 8, instead of 16, does not give significantly different results to the standard case (0.5610, $p = 0.25$). However, using the "patch-up mechanism" throughout, totally replacing the multi-target approach, does lead to a large reduction in consistency (0.4868, $p < 10^{-10}$). This confirms that the simplified "patch-up" approach is good enough, as long as it is not introduced too soon. Explorations of variations in the various learning parameter values have so far failed to identify any significant improvements over the standard network. In particular, varying the output error training tolerance from 0.2 to either 0.1 or 0.3 led to no significant difference in the alignment consistency, and nor did varying the performance level which initiates a new stage of incremental learning from 95% to 99%, 90%, 80% or even 60%.

For comparison, Table 1 also shows the results of a naïve alignment and the Damper et al. alignment [9]. The naïve alignment, that simply assigns one phoneme to each letter from left to right till they run out, not surprisingly results in very low consistency (0.2276). Computing the consistency for the Damper et al. [9] approach is complicated by the fact that they specify their alignments by inserting both "null letters" and "null phonemes". The null phonemes act in the same way as in the neural network approach, but rather than allowing more than one output phoneme per letter, any extra phonemes are aligned with null letters. For example, the word AXE leads to the 3-letter mapping "A X E ⇒ ae k+s _" in the neural network approach, and the 4-letter mapping "A X _ E ⇒ ae k s _" in the Damper et al. approach. Here one can take the "graphemes" *G* to be the letters plus null, and the "phonemes" *F* to be the actual phonemes plus null. That gives a consistency *C* of 0.5560. To get a fairer comparison with the approach of this paper, the null letters can be removed, and the phonemes associated with them



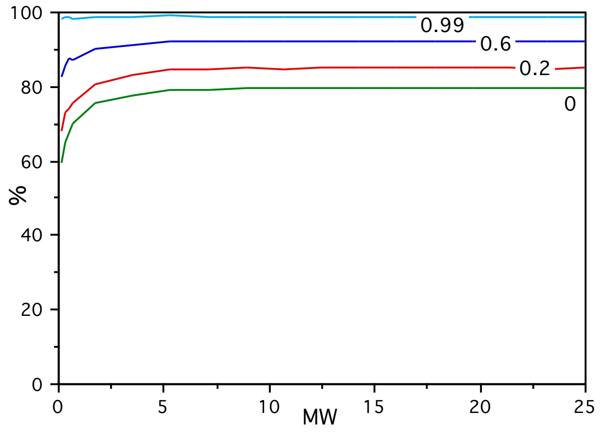


Fig 3. Generalization performance (percentage correct) against number of millions of words of training (MW) for leniency parameter $K = 0, 0.2, 0.6, 0.99$. Two distinct learning patterns arise: the standard incremental approach (left) and the non-incremental equivalent (right).

assigned to adjacent real letters, to match the neural network representation. For a null letter at the beginning or end of a word, it is clear which letter the corresponding phoneme should be associated with. When they appear mid-word, it is not so obvious, but it matches the neural representation most closely if the phoneme is associated with the real letter that comes first. One can then compute the letter to phonemes consistency in the same way as for the neural networks, and that results in the increased value of 0.5630.

So the average consistency achieved by any of the neural networks is still slightly below that of the Damper et al. [9] approach. Indeed, even the best individual network does not do better. One possible reason is that not allowing a letter to map to more than two phonemes is a restriction which the Damper at al. approach doesn't have, and this can cause anomalous alignment problems that result in reduced consistency. This would not be a big problem for standard English, but the BEEP dictionary contains numerous entries that appear to be incorrect pronunciations, e.g.

DISGUSTEDNESS	⇒	d ih s ih n t r ax s t ih d n ax s
IMPROBABLENESS	⇒	ih m p r ae k t ih k ax b l n ax s
RECONGELATION	⇒	r eh k ax n g r ae ch uh l ey sh ax n

and such errors are likely to be difficult to avoid completely in any very large pronunciation dictionary [13]. For such words, it can be impossible with only two phonemes per letter to accommodate the extraneous phonemes at the appropriate position, and that can throw out the alignment for the letters that are actually pronounced correctly. Fortunately, the restriction to two output blocks is not a fundamental limitation of the multi-target learning approach. For example, the same kind of multi-target learning network has been shown to work well mapping one phoneme to up to four letters in a spelling model [4, 6]. However, the number of potential alignments, and hence target outputs, rises rapidly with the number of output blocks, so computational resource limitations prevent this paper from presenting an exploration of the potential improvements that might result from allowing more than two output phonemes per letter.

Another relevant issue is the inherent variance in the neural network outputs that arise from the random factors involved, in particular due to the different random initial weights and different random order of presentation of the training words. However, it has been known for some time now that ensembles of neural networks can be combined together as a voting committee machine to result in improved performance over the average, and sometimes even over the best individual [1, 11]. This approach will work whenever the errors made by individual networks are relatively rare and uncorrelated, so they can be out-voted by the rest of the ensemble. Table 1 shows the consistencies C of the ensemble alignments for the main neural network variations. The ensemble of standard neural networks is best, with an alignment consistency that surpasses all the individual neural network results, and also matches that of the Damper et al. [9] approach (to four decimal places!).

VI. Neural Network Generalization Results

For pronunciation systems, it is the generalization ability that is the ultimate test of performance, i.e. how well the trained networks perform on unseen inputs. Damper et al. [9] used a leave-one-out cross-validation scheme to measure that, but training a new neural network for each word in the training set is not practical. Instead, generalization here was measured using a 10-fold cross-validation approach. The full set of training data was split randomly into 10 sub-sets, and a new network trained on each of the ten combinations of nine sub-sets, and tested on the unseen sub-set.

The generalization performance was typically found to be in the region of 80%, which seems quite low. Damper et al. [9] achieved around 86% using their leave-one-out cross validation approach on a reduced data-set. They had to omit about 10% of the full word set because their alignments involved "null letters" which the PbA system could not accommodate. Those missing words are likely to be the ones that any alignment system will find most difficult to accommodate. In fact, removing those words leads to the Damper et al. [9] alignment consistency C increasing from 0.5630 to 0.5711, which means that reduced word set is

	$K = 0$	$K = 0.2$	$K = 0.4$	$K = 0.6$	$K = 0.8$	$K = 0.9$	$K = 0.99$
Standard NN	80.67	85.40	87.98	92.35	94.85	96.65	98.95
Non-incremental	79.49 *	84.61 *	87.27 *	91.06 *	93.66 *	95.39 *	97.64 *
Flat initial weights	80.54	84.84 *	87.47 *	91.78 *	94.31 *	96.08 *	98.35 *
No MT learning	79.47 *	83.67 *	85.81 *	88.64 *	90.91 *	92.61 *	94.65 *
Aligned std. data	80.86	85.84 *	88.45 *	92.77 *	95.23	97.00	99.37 *
Aligned std., Non-inc	79.77 *	84.96	87.74	92.26	94.92	96.80	99.29
Aligned std., Flat wts	80.67	85.67	88.29	92.58	95.11	96.93	99.30
Naïve aligned data	52.97 *	58.90 *	60.71 *	63.63 *	65.73 *	69.53 *	72.98 *
Damper et al. subset	79.62 *	85.06	87.81	92.51	95.27 *	97.23 *	99.29 *

Table 2. Generalization performances (percentages correct) for the neural network variations and a range of leniency parameters *K*. Values that differ significantly (unpaired two-tailed t-test $p < 0.01$) from the corresponding standard network results are indicated by a "*".

likely to over-estimate the generalization performance on the full word set. Damper et al. [9] note that if all those omitted words were counted as wrong, they would still achieve a generalization performance of 76.1%, which they considered "a very respectable result on such a sizeable dictionary". Of course, no such word removals are needed for the approach of this paper, so it appears that the new neural network generalization results are actually quite good compared to those of the existing symbolic approaches.

Another complication is that the whole concept of correct generalization is not straightforward here. First, the full BEEP data set actually contains multiple pronunciations for many words, and only the first for each word was selected to give a consistent set for training [9]. One could always use the full set for testing purposes, but there is no guarantee that the BEEP dictionary contains the full set of possibilities. In fact, given the range of acceptable regional variations in pronunciation, it seems likely that it is actually far from complete. Moreover, the BEEP dictionary contains a number of items of dubious accuracy [13], and there is little chance of any system generalizing to get those correct.

For the earlier smaller-scale reading models, which were usually trained on a standard set of mono-syllabic words, it was feasible to test them on a standard representative set of hand-crafted non-words, many of which had a range of acceptable pronunciations [6, 14, 16]. Such a hand-crafted approach is obviously never going to be feasible for testing the large-scale systems of interest here, and producing a reliable automatically generated set is effectively what we are trying to do with the neural network model.

One way to proceed is to use the best set of alignment data to identify acceptable generalizations that differ from the test set pronunciation. Clearly, simply taking the highest frequency phoneme for each letter will not be sufficient. First, there is often no clear winner, particularly for the vowels. For example, in the most consistent alignment found, the top three pronunciations for the letter "A" are the phonemes "ae" (30%), "ax" (25%) and "ey" (17%), leaving 28% for the others. Moreover, standard pronunciation rules are then lost, like the length of a vowel sound depending on the presence or absence of a final letter "E", e.g., "PAN" pronounced "p ae n" and "PANE" pronounced "p ey n".

What one can do is look at a sequence of generalization measures, starting with only allowing the pronunciation listed in the test set, and then being increasingly lenient about what variations are allowed. One can truncate the frequency ordered list of phoneme alignments for each letter at the point at which the total coverage (as a proportion of all mappings) is *K*. Clearly, $K = 0$ means only counting as correct the test set pronunciations. Then $K = 0.2$ corresponds to also allowing the most common phoneme for each letter, i.e. a total of up to 26 extra matches. Increasing to $K = 0.8$ allows multiple possibilities for all the vowels and some consonants ("C" and "S"), totaling 51 matches. By $K = 0.99$, only the letter "V" matches a single phoneme, and there are 121 allowable extra matches in total, none of which are obvious errors. Finally, for $K = 1.00$ there are 838 matches, many of which would be considered unacceptable by most English speakers. The number of generalizations counted as correct will clearly increase with *K*, and there is still no guarantee that the choice of variation follows any kind of appropriate pronunciation rule, but it does enable individual users to choose which cut-off they consider most appropriate for their purposes.

Figure 3 shows how a range of generalization measures progressed during training for the standard incremental approach and the non-incremental equivalent. Having access to the whole training data set at once does lead to a faster improvement in performance for all values of the leniency parameter *K*, but by the end of training, the incremental approach is performing better, and the total number of training word presentations needed to reach the stopping criterion is fewer.

Table 2 presents the final generalization performances for the key neural network variations using a range of values for the leniency parameter *K*. The standard network achieves the best results on non-aligned training data for all values of *K*. For the non-incremental learning variation, there is a small (~1.0%) but significant reduction in performance compared to the standard approach. The variation using a flat initial weight distribution results in an even smaller performance reduction (~0.5%) compared to the standard approach, but the difference is still significant for all values of K except 0. If the full multi-target learning approach is replaced by the faster "patch-up mechanism", there are significant bigger and more variable reductions (>1.2%). Not surprisingly, the equivalent networks trained with the naïve aligned data have the worst results, with very large (>25%) and highly significant reductions compared with the standard network. The various generalization performance reductions are in line with the reductions in alignment consistency seen in Table 1, providing further confidence that the consistency measure C really is a useful and reliable indicator of the alignment quality and of the level of generalization performance that can be expected to result from it.

A remaining important question is whether the standard multi-target learning neural network performs as well as an equivalent neural network learning from the best pre-aligned training data throughout. This was tested by carrying out the training using the best alignment previously achieved, namely that from the ensemble of 20 standard networks. Table 2 shows that there is only a slight generalization improvement (~0.4%) to be achieved in that way, and the difference is only significant for four of the seven K values tested, and not including the pure test-set ($K = 0$) case. Using the incremental learning approach is still crucial for the pre-aligned version. Without it, the standard network is then significantly better for the $K = 0$ case, and for other values of K is not significantly different. If pre-aligned training data is used with a flat distribution of initial weights, there are no significant differences to the standard network results. So any improvements from pre-alignment are lost due to non-incremental learning or flat initial weights.

Unfortunately, it is not possible to test the full Damper et al. [9] alignment by training on a neural network, because of the significant number of words it has which include null letters. As noted above, to accommodate those words in the neural network representation requires the null letters to be removed and the associated phonemes treated as additional phoneme outputs for the adjacent real letters. However that results in a number of letters that are mapped to more than two phonemes and cannot be accommodated by the two output blocks of the neural network. If those (3625) words are simply removed from the data set, the Damper et al. [9] alignment consistency C increases from 0.5630 to 0.5685. Nevertheless, as seen in Table 2, the generalization results are mixed: significantly worse (~1.0%) for the $K = 0$ case, and significantly better (~0.4%) for $K \geq 0.8$. If one were less generous and counted all the removed words as wrong, that would reduce the Damper et al. percentages in Table 1 by a factor of 0.9818, which renders them significantly worse than the standard neural network for all values of K.

VII. Conclusions

This paper has shown how earlier neural network models of reading [3, 5, 6] can be scaled up to cope with the much larger dictionaries and word lengths required for modern speech technology. The resulting text to phoneme alignment consistencies and pronunciation generalization performances have been demonstrated to be at least comparable to the existing state-of-the-art symbolic rule-based systems [9, 10]. Moreover, the networks automatically learn the alignment and mapping simultaneously, and variations in the neural network approach (such as non-standard initial weight distributions, incremental training regimes, computational speed-ups, and different parameter values) have been explored, with the best approaches identified. It has also been demonstrated how a simple neural network ensemble approach [1, 11] can lead to even better performance than the individual networks.

The ideas and neural networks presented here can now be applied fairly straightforwardly elsewhere, in particular, to the reverse mapping (i.e. from phonemes to graphemes as required for spelling), to alternative English dictionaries, and to other languages.

References

[1] Battiti, R. & Colla, A.M. (1994). Democracy in neural networks: Voting schemes for classification. *Neural Networks,* **7**, 691-709.

[2] Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press.

[3] Bullinaria, J.A. (1993). Neural network models of reading multi-syllabic words. In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN 1993),* 283-286. Piscataway, NJ: IEEE.

[4] Bullinaria, J.A. (1994). Connectionist modelling of spelling. In: *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society,* 78-83. Hillsdale, NJ: Erlbaum.

[5] Bullinaria, J.A. (1995). Neural network learning from ambiguous training data. *Connection Science*, **7**, 99-122.

[6] Bullinaria, J.A. (1997). Modelling reading, spelling and past tense learning with artificial neural networks. *Brain and Language,* **59**, 236-266.

[7] Bullinaria, J.A. (2003). Evolving efficient learning algorithms for binary mappings. *Neural Networks*, **16**, 793-800.

[8] Damper, R.I., Marchand, Y., Adamson, M.J. & Gustafson, K. (1999). Evaluating the pronunciation component of text-to-speech systems for English: A performance comparison of different approaches. *Computer Speech and Language*, **13**, 155-176.

[9] Damper, R.I., Marchand, Y., Marsters, J.D.S. & Bazin, A.I. (2005). Aligning text and phonemes for speech technology applications using an EM-like algorithm. *International Journal of Speech Technology*, **8**, 149-162.

[10] Davel, M. & Barnard, E. (2008). Pronunciation predication with Default&Refine. *Computer Speech and Language*, **22**, 374–393.

[11] Hansen, L.K. & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**, 993-1000.

[12] Hare, M. & Elman, J.L. (1995). Learning and morphological change. *Cognition*, **56**, 61-98.

[13] Martirosian, O.M. & Davel, M. (2007). Error analysis of a public domain pronunciation dictionary. In *Proceedings of the 18th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA*), 13-16.

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

[15] Robinson, A. (1997). *British English Example Pronunciation Dictionary (BEEP),* Version 1.0, Cambridge University.

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

[17] Sejnowski, T.J. & Rosenberg, C.R. (1987). Parallel networks that learn to pronounce English text. *Complex Systems*, **1**, 145-168.

[18] Wolff, M., Eichner, M. & Hoffmann, R. (2002). Measuring the quality of pronunciation dictionaries. In *Proceedings of the ISCA Tutorial and Research Workshop on Pronunciation Modeling and Lexicon Adaptation (PMLA),* 117-122.