Abstract

The Cambridge University (Cambridge University) effect from the psycholinguistics literature has demonstrated a robust word processing mechanism in humans, where jumbled words (e.g. Cambridge / Cmabrigde) are recognized with little cost. Inspired by the findings from the Cambridge University effect, we propose a word recognition model based on a semi-character level recursive neural network (scRNN). In our experiments, we demonstrate that scRNN has significantly more robust performance in word spelling correction (i.e. word recognition) compared to existing spelling checkers. Furthermore, we demonstrate that the model is cognitively plausible by replicating a psycholinguistics experiment about human reading difficulty using our model.

1 Introduction

Despite the rapid improvement in natural language processing by computers, humans still have advantages in situations where the text contains noise. For example, the following sentences, introduced by a psycholinguist (Davis, 2003), provide a great demonstration of the robust word recognition mechanism in humans.

Aoccdrnig to a rscheearch at Cmabrigde Universtiy, it deosn’t mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rght pclae. The reet can be a toatl mses and you can stil raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed er-vey lteter by istlef, but the wrod as a wlohe.

This example shows the Cambridge University (Cambridge University) effect, which demonstrates that human reading is resilient to (particularly internal) letter transposition.

Robustness is important and useful property for various NLP tasks, and we propose a computational model which replicates this robust word recognition mechanism. The model is based on a standard recurrent neural network with a memory cell as in LSTM (Hochreiter and Schmidhuber, 1997). The input layer of the model consists of three sub-vectors: beginning (b), internal (i), and ending (e) character(s) of the input word (Figure 1). This semi-character level recurrent neural network is referred as scRNN.

First, we review previous work on the robust word recognition mechanism from psycholinguistics literature (§2). Next, we describe technical details of scRNN which capture the robust human mechanism (§3) using recent developments in neural networks. Our experiments show that the scRNN outperforms commonly used spelling checkers by 42% for jumbled word correction (§4.1) and 5% and 27% in other noise types (insertion and deletion). We also show that scRNN replicates recent findings from psycholinguistics experiments on reading difficulty (i.e. accuracy) depending on the position of jumbled letters, which indicates that scRNN successfully mimics
The boy could not solve the problem so he asked for help.

The boy could not solve the problem so he asked for help.

<table>
<thead>
<tr>
<th>Cond.</th>
<th>Example</th>
<th># of fixations</th>
<th>Regression(%)</th>
<th>Avg. Fixation (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>The boy could not solve the problem so he asked for help.</td>
<td>10.4</td>
<td>15.0</td>
<td>236</td>
</tr>
<tr>
<td>INT</td>
<td>The boy could not solve the problem so he asked for help.</td>
<td>11.4*</td>
<td>17.6*</td>
<td>244*</td>
</tr>
<tr>
<td>END</td>
<td>The boy could not solve the problem so he asked for help.</td>
<td>12.6\†</td>
<td>17.5\†</td>
<td>246*</td>
</tr>
<tr>
<td>BEG</td>
<td>The boy could not solve the problem so he asked for help.</td>
<td>13.0\‡</td>
<td>21.5\‡</td>
<td>259\†</td>
</tr>
</tbody>
</table>

Table 1: Example sentences and results for measures of fixation (excerpt from (Rayner et al., 2006)). There are 4 conditions: N = normal text; INT = internally jumbled letters; END = letters at word endings are jumbled; BEG = letters at word beginnings are jumbled. Entries with \* have statistically significant difference from the condition N (p < 0.01) and those with \† and \‡ differ from \* and \† with p < 0.01 respectively.

(at least a part of) the human word recognition mechanism (§4.2).

2 Reading Words with Jumbled Letters

Sentence processing with jumbled words has been a major research topic in psycholinguistics literature. Forster et al. (1987) show that a jumbled word (e.g. answer-ANSWER) facilitates primes as large as identity primes (answer-ANSWER) in a masked priming paradigm and these results have been confirmed (Perea and Lupker, 2004; Guer- rera and Forster, 2008).

These findings about robust word processing mechanism by human have been further investigated by looking at other types of noise in addition to simple letter transpositions. Humphreys et al. (1990) show that deleting a letter in a word still produces significant priming effect (e.g. black-BLACK), and similar results have been shown in other research (Peressotti and Grainger, 1999; Grainger et al., 2006). Van Assche and Grainger (2006) demonstrate that a priming effect remains when inserting a character into a word (e.g. justice-JUSTICE).

With an eye-movement paradigm, Rayner et al. (2006) and Johnson et al. (2007) conduct more detailed experiments on the robust word recognition mechanism with jumbled letters. They show that letter transposition affects fixation time measures during reading depending on which part of the word is jumbled. Table 1 presents the result from Rayner et al. (2006). It is obvious that human can read smoothly (i.e. smaller number of fixa- tions, regression, and average of fixation duration) when a given sentence has no noise (referred this condition as N). When the characters at the begin- ning of word are jumbled (referred this condition as BEG), human have more difficulty. The other two conditions, where words are internally jumbled (INT) or letters at word endings are jumbled (END), have similar amount of effect, although the number of fixations between them showed a statistically significant difference (p < 0.01). In short, the reading difficulty with different jumble conditions is summarized as follows: N < INT ≤ END < BEG.

It may be surprising that there is statistically significant difference between END and BEG conditions despite the difference being very subtle (i.e. fixing either the first or the last character). This result demonstrates the importance of beginning letters for human word recognition.\footnote{There is still a debate in psycholinguistics community whether or not the order of internal letters does not matter at all. In this paper, we keep the original hypothesis in order for our model to be simple.}

3 Semi-Character Recurrent Neural Net

In order to achieve the human-like robust word processing mechanism, we propose a semi-character based recurrent neural network (scRNN). The scRNN has the same structure as a standard recurrent neural network except that the input vector consists of three sub-vectors that correspond to the characters’ position. The first and third sub-vectors ($b_n, e_n$) represent the first and last character of the $n$-th word. These two sub-vectors are therefore one-hot representations. The second sub-vector ($i_n$) represents a bag of characters of the word not including the initial and final positions. For example, the word “University” is represented as $b_n = \{U = 1\}, e_n = \{y = 1\}$, and $i_n = \{e = 1, i = 2, n = 1, s = 1, r = 1, t = 1, v = 1\}$, with all the other elements being zero. The size of sub-vectors ($b_n, i_n, e_n$) is equal to the number of characters ($N$) in our language, and $x_n$ has therefore the size of $3N$ by concatenating the sub-vectors.

With this input vector ($x_n$), the recurrent neural
According to a researcher at Cambridge University, it does not matter in what order the letters in a word are, the only important thing is that the first and last letter be at the right place. The rest can be a total mess and you can still read it without problem. This is because the human mind does not read every letter by itself, but the word as a whole.

| Original | Aoccdrnig to a rscheearch at Cambrigde Uinervtisy, it deos n’t mttier in wahht oredr the letteers in a wrod are, the olny iprmoennt thiing is taht the frist and lsat letteer be at the rghit pclae. The rest can be a toatl mses and you can stll raed it wouthit porbelm. Thiis hiuscaes the huamn mnid dos not raed ervry letteer by istlef , but the wrod as a whole. |
| Correct | According to a researcher at Cambridge University, it does not matter in what order the letters in a word are, the only important thing is that the first and last letter be at the right place. The rest can be a total mess and you can still read it without problem. This is because the human mind does not read every letter by itself, but the word as a whole. |
| scRNN | According to a researcher at Cambridge University, it does not matter in what order the letters in a word are, the only important thing is that the first and last letter be at the right place. The rest can be a total mess and you can still read it without problem. This is because the human mind does not read every letter by itself, but the word as a whole. |
| Commercial | According to a researcher at Cambridge University, it does not matter in what order the letters in a word are, the only important thing is that the first and last letter be at the right place. The rest can be a total mess and you can still read it without problem. This is because the human mind does not read every letter by itself, but the word as a whole. |
| Hunspell | According to a researcher at Cambridge University, it does not matter in what order the letters in a word are, the only important thing is that the first and last letter be at the right place. The rest can be a total mess and you can still read it without problem. This is because the human mind does not read every letter by itself, but the word as a whole. |

Table 2: Example spelling correction outputs for the Cmabrigde Uinervtisy sentences. Underline indicates words which the system failed to correct.

The input layer is used as input to a hidden layer \( h_n \) which is an LSTM. The LSTM layer has a recurrent input from its previous state \( h_{n-1} \). The output of the hidden layer is taken as input to the softmax function layer, which predicts an output word \( y_n \). We use the cross-entropy training criterion applied to the output layer as in most LSTM language modeling works; the model learns the weight matrices \( W_{hh}, W_{hy} \) to maximize the likelihood of the training data. This should approximately correlate with maximizing the number of exact word match in the predicted outputs. Figure 1 shows a pictorial overview of scRNN.

In order to check if the scRNN can recognize the jumbled words correctly, we test it in spelling correction experiments. If the hypothesis about the robust word processing mechanism is correct, scRNN will also be able to read sentences with jumbled words robustly.

### 4 Experiments

We conducted spelling correction experiments to judge how well scRNN can recognize noisy word sentences under different conditions. We used Penn Treebank for training, tuning, and testing.\(^2\) The input layer consists of a vector with length of 76 (A-Z, a-z and 24 symbol characters). The hidden layer units had size 650, and total vocabulary size was set to 10k. We apply one type of noise to every word except that all words with numbers (e.g. 1980s) and short words (length \( \leq 3 \)) are not subjected to jumbling and were left as is, and therefore these words are excluded in evaluation. We trained the model by running 5 epochs with batch size 20. In order to make the training efficient, we set the backpropagation through time (BPTT) parameter to 3.

#### 4.1 Spelling correction results

We tested different noise types: jumble, delete, and insert, where the jumble changes the internal characters (e.g. Cambridge \( \rightarrow \) Cmbarigde), delete randomly deletes one of the internal characters (Cambridge \( \rightarrow \) Cambridge), and insert randomly inserts an alphabet into an internal position (Cambridge \( \rightarrow \) Cambbridge). None of the noise types change the first and last characters. For comparison, we ran two widely-used spelling checkers (Commercial\(^3\) and Hunspell\(^4\)).

Table 2 presents example outputs for the Cmabrigde Uinervtisy sentence by each model.\(^5\) It is clear that scRNN performs better than the

---

\(^2\) Section 2-21 for training, 22 for tuning, and 23 for test.

\(^3\) We anonymized the name of the commercial product.

\(^4\) https://hunspell.github.io/

\(^5\) The Cmabrigde Uinervtisy sentences contains jumbling as well as deletion, insertion, and replacement of characters.
Table 3: Spelling correction accuracy (%) with different error types. The difference between scRNN and the other two models are statistically significant ($p < 0.001$).

Other spelling checkers. The only error in scRNN may be because the last character (rscheearc) activated the scRNN nodes strongly toward research instead of researcher.\footnote{There is also a deletion of ‘r’.

Table 4 shows the result with respect to noise type. Overall, scRNN outperforms the other two spelling checkers across all three different noise types. It is striking that scRNN shows robustness in jumble noise, whereas the other models are significantly affected. All three models suffer under the delete condition, but scRNN still maintains 85% of accuracy whereas the other models decreased to 58% and 35% accuracy.

The relatively large drop in delete in scRNN may be because the information lost by deleting character is significant. For example, when the word place has dropped the character l, the surface form becomes pace, which is also a valid word. Also, the word mess with e being deleted produces the form of mss, which can be recovered as mess, mass, miss, etc. In the Cambridge University sentences, in both cases, the local context support other phrase such as ‘at the right pace/place’ and ‘a total mass/mess’. These examples clearly demonstrate that deleting characters harm the word recognition more significantly than other noise types.

Finally, all the three models perform well on insert noise, indicating that adding extraneous information by inserting a letter does not change the original information significantly.

4.2 Corroboration with psycholinguistic experiments

As seen in §2, the position of transposition affects the cognitive load of human word recognition. We investigate this phenomenon with scRNN by manipulating the structure of input vector. We replicate the experimental paradigm in Rayner et al. (2006), but using scRNNs rather than human subjects. We trained the scRNN on different jumble conditions: INT = internal letters are jumbled; END = letters at word endings are jumbled; BEG = letters at word beginnings are jumbled; ALL = all letters are jumbled. Entries with * have statistically significant difference from the condition INT ($p < 0.001$) and those with † and ‡ differ from * and † with $p < 0.001$ respectively.

Table 4: Spelling correction accuracy with 4 different jumble conditions: INT = internal letters are jumbled; END = letters at word endings are jumbled; BEG = letters at word beginnings are jumbled; ALL = all letters are jumbled. Entries with * have statistically significant difference from the condition INT ($p < 0.001$) and those with † and ‡ differ from * and † with $p < 0.001$ respectively.

5 Related Work

Character-based recurrent neural networks have been investigated and used for a variety of NLP tasks such as language modeling (Sutskever et al., 2011), segmentation (Chrupała, 2013), dependency parsing (Ballesteros et al., 2015), machine translation (Ling et al., 2015), and text normalization (Chrupała, 2014). Although scRNN has some similarity in terms of model architecture with these recent works, our contribution is the demonstration of the robustness and cognitive plausibility of semi-character-based recurrent neural networks for word recognition.

Character-level convolutional neural nets (CNN) have also been noted (Kim et al., 2015) and used for spelling correction (Schmaltz et al.,
While CNNs have a richer representation, scRNN still achieves high accuracy in jumbled word recognition with a simpler RNN structure (i.e. no convolution layers are required) resulting in fast training time and small model size.

### 6 Summary

We have presented a semi-character recurrent neural network model, scRNN, which is inspired by the robust word recognition mechanism known in psycholinguistics literature as the *Cambridge University* effect. Despite the model’s simplicity, it significantly outperforms existing spelling checkers with respect to various noise types. We also have demonstrated a similarity between scRNN and human word recognition mechanisms, by showing that scRNN replicates a psycholinguistics experiment about word recognition difficulty in terms of the position of jumbled characters.

There are a variety of potential NLP applications for scRNN where robustness plays an important role, such as normalizing social media text (e.g. `Coooolll` → `Cool`) and modeling morphologically rich languages.

### References


