Neural Transduction of Letter Position Dyslexia using an Anagram Matrix Representation

Avi Bleiweiss
BShalem Research
Sunnyvale, CA, USA
avibleiweiss@bshalem.onmicrosoft.com

Abstract

Research on analyzing reading patterns of dyslectic children has mainly been driven by classifying dyslexia types offline. We contend that a framework to remedy reading errors in-line is more far-reaching and will help to further advance our understanding of this impairment. In this paper, we propose a simple and intuitive neural model to reinstate migrating words that transpire in letter position dyslexia, a visual analysis deficit to the encoding of character order within a word. Introduced by the anagram matrix representation of an input verse, the novelty of our work lies in the expansion from one to a two dimensional context window for training. This warrants words that only differ in the disposition of letters to remain interpreted semantically similar in the embedding space. Subject to the apparent constraints of the self-attention transformer architecture, our model achieved a unigram BLEU score of 40.6 on our reconstructed dataset of the Shakespeare sonnets.

1 Introduction

Dyslexia is a reading disorder that is perhaps the most studied of learning disabilities, with an estimated prevalence rate of 5 to 17 percentage points of school-age children in the US (Shaywitz and Shaywitz, 2005; Made by Dyslexia, 2019). Counter to popular belief, dyslexia is not only tied to the visual analysis system of the brain, but also presents a linguistic problem and hence its relevance to natural language processing (NLP). Dyslexia manifests itself in several forms as this work centers on Letter Position Dyslexia (LPD), a selective deficit to encoding the position of a letter within a word while sustaining both letter identification and character binding to words (Friedmann and Gvion, 2001).

A growing body of research advocates heterogeneity of dyslexia causes to poor non-word and irregular-word reading (McArthur et al., 2013). Along the same lines Kezilas et al. (2014) suggest that character transposition effects in LPD are most likely caused by a deficit specific to coding the letter position and is evidenced by an interaction between the orthographic and visual analysis stages of reading. To this end, more recently Marcet et al. (2019) managed to significantly reduce migration errors by either altering letter contrast or presenting letters to the young adult sequentially.

To dyslectic children not all letter positions are equally impaired as medial letters in a word are by far more vulnerable to reading errors compared to the first and last characters of the word (Friedmann and Gvion, 2001). Children with LPD have high migration errors where the transposition of letters in the middle of the word leads to another word, for example, *slime–smile* or *cloud–could*. On the other hand, not all reading errors in cases of selective LPD are migratable and are evidenced by words read without a lexical sense e.g., *slime–silme*. Intriguingly, increasing the word length does not elevate the error rate, and moreover, shorter words that have lexical anagrams are prone to a larger proportion of migration errors compared to longer words that possess no-anagram words. A key observation for LPD is that although words read may share all letters in most of the positions, they still remain semantically unrelated.

Machine learning tools to classify dyslexia use a large corpus of reading errors for training and mainly aim to automate and substitute diagnostic procedures expensively managed by human experts. Lakretz et al. (2015) used both LDA and Naive Bayes models and showed an area under curve (AUC) performance of about 0.8 that exceeded the quality of clinician-rendered labels. In their study, Rello and Ballesteros (2015) proposed a statistical model that predicts dyslectic readers using eye tracking measures. Employing an SVM-based binary classifier, they achieved about 80% accuracy.

Proceedings of the BioNLP 2020 workshop, pages 150–155

Instead, our approach applies deep learning to the task of restoring LPD inline that we further formulate as a sequence transduction problem. Thus, given an input verse that contains shuffled-letter words identified as transpositional errors, the objective of our neural model is to predict the originating unshuffled words. We use language similarity between predicted verses and ground-truth target text-sequences to quantitatively evaluate our model. Our main contribution is a concise representation of the input verse that scales up to moderate an exhaustive set of LPD permutable data.

2 Anagram Matrix

Using a colon notation, we denote an input verse to our model as a text sequence $w_{1:n} = (w_1, \ldots, w_n)$ of n words interchangeably with n collections of letters $l_{1:n} = (l^{(1)}_{1:|w_1|}, \ldots, l^{(n)}_{1:|w_n|})$. We generate migrated word patterns synthetically by anchoring the first and last character of each word and randomly permuting the position of the inner letters $(l^{(1)}_{2:|w_1|-1}, \ldots, l^{(n)}_{2:|w_n|-1})$. Thus given a word with a character length $|l^{(i)}|$, the number of possible unique transpositions for each word follows $t_{1:n} = (|l^{(1)}|!, \ldots, |l^{(n)}|!)$. Next, we extract a migration amplification factor $k = \text{argmax}_{i=1}^{n}\, t_i$ that we apply to each word in an input verse independently and form the sequence $m_{1:k} = (m_1, \ldots, m_k)$. Word length commonly used in experiments of previous LPD studies averages five letters and ranges from four to seven letters long, hence migrating to feasible 2, 6, 24, and 120 letter substitutions, respectively. We note that words with 1, 2, or 3 letters are held intact and are not migratable.

when	forty	winters	shall	besiege	thy	brow
wehn	fotry	wenitrs	sahll	bseeige	thy	borw
when	froty	winrtes	slhal	begseie	thy	borw
wehn	fotry	wrenits	slahl	begisee	thy	borw
when	forty	wtenirs	shall	begeise	thy	brow
when	froty	wtneirs	shall	bigeese	thy	brow
when	ftory	weinrts	sahll	bgiesee	thy	borw
wehn	frtoy	wirtens	slhal	bisgeee	thy	borw
wehn	froty	wterins	slahl	beeisge	thy	brow
when	froty	wtiners	shlal	beesgie	thy	borw
wehn	frtoy	wnetris	shall	beisege	thy	borw

Table 1: A snapshot of letter-position migration patterns in the form of an anagram matrix. The unedited version of the text sequence is highlighted on top.

To address the inherent semantic unrelatedness between transpositioned words, we define a two-dimensional migration-verse array in the form of an anagram matrix $A = [m^{(1)}_{1:k}; , \ldots, ; m^{(n)}_{1:k}] \in \mathbb{R}^{k \times n}$, where $m^{(i)}$ are column vectors, $[\cdot;\cdot]$ is column-bound matrix concatenation, and k and n are the transposition and input verse dimensions, respectively. In Table 1, we render a subset of an anagram matrix drawn from a target verse with a maximal word length of seven letters. The anagram matrix founds an effective context structure for a two-pass embedding training, and our training dataset thus reconstructs on the basis of a collection of anagram matrices with varying dimensions.

3 LPD Embeddings

Models for learning word vectors train locally on a one-dimensional context window by scanning the entire corpus (Mikolov et al., 2013). Through evaluation on a word analogy task, these models capture linguistic regularities as linear relationships between word embeddings. Mikolov et al. (2013) proposed the skip-gram and continuous-bag-of-words (CBOW) neural architectures with the objective to predict the context of the target word and the target word given its context, respectively. Notably LPD migrating words tend mostly outside the English vocabulary and thus pretrained word embeddings on large corpora are of limited use in our system. [1]

Figure 1: A two-dimensional context window of size two drawn from outside context cells of an anagram matrix. The center words are shown in gray for both the normal by-row $\{w_{t,t-2}, \ldots, w_{t,t+2}\}$ and transposed column-wise $\{w_{t-2,t}, \ldots, w_{t+2,t}\}$ forms of feeding our neural network.

While the essence of our task is formalized as verse simplification, mending LPD relies on robust discovery of word similarities along both the migration and verse axes of the anagram matrix. To this extent, we reshape the context window to train word embeddings from one to a two-dimensional array. In Figure 1, we show a bi-dimensional con-

[1] https://nlp.stanford.edu/projects/glove/

text window of size two that is a visible subset drawn from outside context cells of an anagram matrix. Learning word vectors for LPD is a two-pass process in our model. First, the context window W feeds our neural network row-by-row for each transpositioned verse, and then follows by iterating migration vectors $m^{(i)}$ in W^T as inputs.

4 Model

Our task is inspired by recent advances in neural machine translation (NMT). NMT architectures have shown state-of-the-art results in both the form of a powerful sequence model (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2015) and more recently, the cross-attention ConvS2S (Elbayad et al., 2018) and the self-attention based transformer (Vaswani et al., 2017) networks. Given an unintelligible diction of shuffled-letter words, our model aims to output a verse that preserves the semantics of the input, and uses the transformer that outperforms both recurrent and convolutional configurations on many language translation tasks.

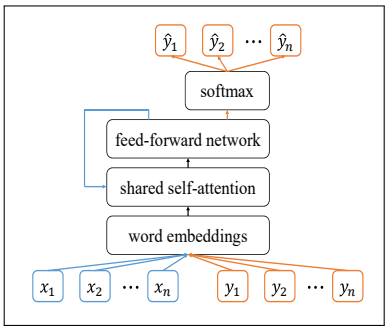

Figure 2: Transformer architecture overview (encoder path shown in blue, decoder in brown).

Stacked with several network layers, the transformer architecture only relies on attention mechanisms and entirely dispensing with recurrence (Hochreiter and Schmidhuber, 1997; Chung et al., 2014). In Figure 2, we show a synoptic rendition of the transformer. Its inputs consist of a source verse with potentially letter-transpositioned words x_i, and a ground-truth target verse of words with unshuffled letters y_i. The transformer encoder and decoder modules largely operate in parallel and provide for a source-to-target attention communication, and a softmax layer operates on the decoder hidden-state outputs to produce predicted words $\hat{y}_i$. In LPD, source and target verses are consistently of the same word count n, however, copying tokens from the source over to predictions is inconsequential to the quality of repairing reading errors due to extensive out-of-vocabulary non-migrating words.

5 Setup

To quantitatively evaluate our LPD transduction approach, we chose to mainly report n-gram BLEU precision (Papineni et al., 2002) that defines the language similarity between a predicted text sequence and the ground-truth reference verse. In the BLEU metric, higher scores indicate better performance.

5.1 Corpus

Rather than clinical reading tests, we used the Sonnets by William Shakespeare (Shakespeare, 1997). This is motivated by the apostrophe-rich data that forces left-out letters. The raw dataset comprises 2,154 verses that range from four to fifteen word sequences. In Figure 3, we show the distribution of word length across the dataset, as 18,858 unique tokens are of up to seven-letter long inclusive and take about 62 percentage points of the entire corpus words. To conform to preceding LPD research, we conducted a cleanup step that removes all words of eight letters or more from the dataset. We hypothesize that evaluating LPD on a single word basis lets us perform this step without loss of generality.

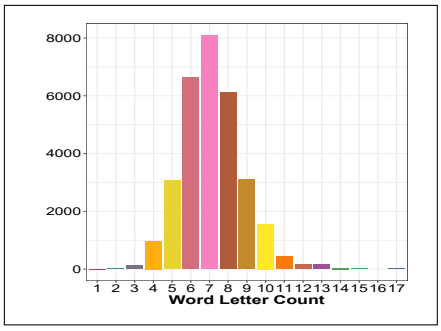

Figure 3: Distribution of word letter count across unique tokens of the Shakespeare Sonnets dataset.

We then transform each verse of the Sonnets to an anagram matrix representation A. The verse word with the maximal letters has a set of distinct traspositions while words of lesser letters are shuffled with repetition (Table 1). In Figure 4, we show the distribution of anagram matrices across the entire Shakespeare Sonnets dataset, with a migration amplification factor $k \in \{1, 2, 6, 24, 120\}$ and a cleaned up verse that spans two to thirteen words. Evidently most prominent tiles are of words with seven letters and consist of verse sizes between seven to nine words. Concatenating the rows of all

the anagram matrices presents a sixtyfold extended shape of our LPD training dataset that has 130,021 text sequences, along with source and target vocabularies of 173,575 and 3,147 tokens, respectively.

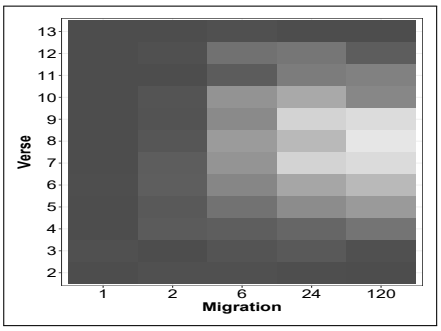

Figure 4: Distribution of anagram matrices across the verse collection of the Shakespeare Sonnets dataset.

5.2 Training

We used PyTorch (Paszke et al., 2017) version 1.0 as our deep learning platform for training and inference. PyTorch supports the building of effective neural architectures for NLP task development.

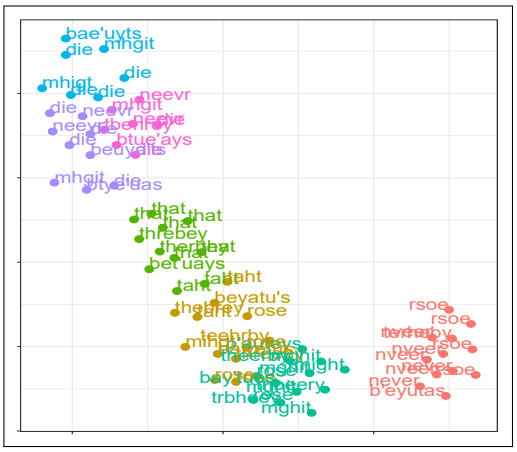

Figure 5: Aided by using our anagram matrix approach, migrated and non-migrated embeddings shown to preserve unedited input similarity. Presented in seven clusters produced by k-means (R Core Team, 2013).

We incorporated in our framework the annotated PyTorch implementation of the transformer (Rush, 2018) and modified it to accommodate our LPD dataset. Multi-head attention was configured with $h = 8$ layers and a model size $d_{model} = 512$, and the query, key, and value vectors were set uniformly to $d_{model}/h = 64$. The inner layer of the encoder and the decoder had dimensionality $d_{ff} = 2,048$. In Figure 5, we show permuted embeddings retaining input semantics by using our anagram matrix concept. The presence of replicated words in vector space owes to the transformer built-in learned positions of input embeddings. We chose the Adam optimizer (Kingma and Ba, 2014) with a varied learning rate and a fixed model dropout of 0.1, using cross-entropy loss and label smoothing for regularization. Figure 6 reviews epoch-loss progression in training and validating our model.

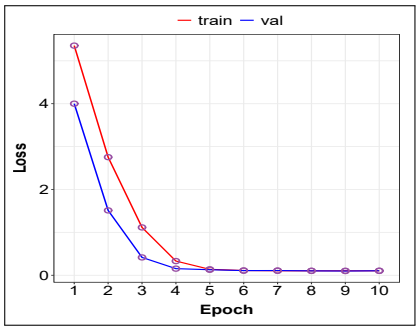

Figure 6: Epoch-loss progression in training and validation. Loss descent subsides near the seventh epoch.

6 Results

We ran our model inference on a split test set that comprises randomly selected rows sampled from the entire collection of anagram matrices and further excluded from the train set. We postulate that the use of matrix columns along the migration axis are only beneficial for embedding training.

Context Window	BLEU-1	BLEU-2	BLEU-3	BLEU-4
one-dimensional	36.8	20.9	13.0	8.3
two-dimensional	**40.6**	**23.7**	**14.7**	**8.9**

Table 2: Model performance using n-gram BLEU measures at a corpus level on our augmented Sonnets testset for repairing letter transpositions. Scores shown are contrasted between the use of one and two dimensional context window for training word embeddings.

In Table 2, we report corpus-level n-gram BLEU scores of our transformer-based model for inline transduction of LPD reading patterns. Uniformly a two-dimensional context window for training embeddings boosts our performance by about ten percentage points on average compared to the one-dimensional window. As expected, BLEU scores decline exponentially when we increase n-gram, from 40.6 for BLEU-1 down to 8.9 for BLEU-4.

While BLEU scores the output by counting n-gram matches with the reference, we also evaluated our model using SARI (Xu et al., 2016), a novel

metric that correlates with human judgments and designed to specifically analyze text simplification models. SARI principally compares system output against both the reference and input verse and returns an arithmetic average of n-gram precisions and recalls of addition, copy, and delete rewrite operations. [2] Table 3 summarizes SARI and average BLEU measures of our model. Scores appear fairly correlated with a slight edge in favor of SARI that correctly rewards models like ours which make changes that simplify input verses.

Context Window	SARI	BLEU
one-dimensional	21.2	19.8
two-dimensional	**23.7**	**22.0**

Table 3: Model performance using automatic evaluation measures of SARI and BLEU at a corpus level on our augmented Sonnets test-set. Scores are contrasted between the use of one and two dimensional context window for training word embeddings.

The transformer is known to be bound by a fixed-length context and thus tends to split a long context to segments that often ignore semantic boundaries. This led to the conjecture that context fragmentation may impact our model performance adversely. The novel transformer-xl network (Dai et al., 2019) that learns dependencies across subsequences using recurrence, might be the more effective architecture to perform our task.

7 Discussion

To conduct a baseline evaluation of our model, we hand curated a corpus made of LPD screening tests. Targeted screeners are brief performance measures intended to classify at-risk individuals. To the extent of our knowledge, Lakretz et al. (2015) used for their experiments the largest known screener dataset to date that consisted of 196 loose target words in Hebrew. Correspondingly, we assembled a screening corpus of 196 English words that are prone to erroneous reading. In our system, these words are recast into a set of anagram matrices, each however reduced to a vector $\in \mathbb{R}^{k\times 1}$. Further downstream, we represented context-less words as one-hot vectors. As expected, on the task of reinstating screener data our sequence model achieved a fairly low 1-gram BLEU score of 9.2. Counter to nearly 4.4X improvement on the Sonnets dataset, when trained using a 2D context window.

[2] https://github.com/cocoxu/simplification

Compared to almost two orders of magnitude larger Sonnets dataset, the screening corpus was too small and thus overfitting our transformer-based neural model. In addition, to effectively exploit our proposed anagram matrix representation, rather than disjoint words we require to train our sequence model on a dataset comprised of verses or sentences that provides essential context for learning embeddings.

In a practical application framework, our proposed model is rated on successful recovery from LPD reading errors that transpire in a text sequence. We envision our model already pretrained on multiple corpora, each extended to a collection of anagram matrices. Every editing instance follows with a dyslectic individual who reads and utters a verse at a time from a text document. Fed to the network, the verse is then inferred by our model that returns an amended text sequence the user can compare side-by-side on his display. It is key for the system we presented to perform responsively.

8 Conclusions

In this paper, we presented word-level neural sentence simplification to aid letter-position dyslectic children. We modeled the task after a monolingual machine translation and showed the representation effectiveness of a two-dimensional context window to boost our model performance. Future avenues of research include using our model in real-world restoration scenarios of LPD, and exploring the efficacy of the transformer-xl architecture to a non language modeling task like ours. We look forward to leverage the exceptional ability of transformer-xl to perform character-level language modeling and improve mending LPD.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful suggestions and feedback.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations, (ICLR)*, San Diego, California.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder

for statistical machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555. Http://arxiv.org/abs/1412.3555.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2978–2988, Florence, Italy.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2018. Pervasive attention: {2D} convolutional neural networks for sequence-to-sequence prediction. In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CONLL)*, pages 97–107, Brussels, Belgium.

Naama Friedmann and Aviah Gvion. 2001. Letter position dyslexia. *Journal of Cognitive Neuropsychology*, 18(8):673–696.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yvette Kezilas, Saskia Kohnen, Meredith Mckague, and Anne Castles. 2014. The locus of impairment in english developmental letter position dyslexia. *Frontiers in human neuroscience*, 8:1–14.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980. http://arxiv.org/abs/1412.6980.

Yair Lakretz, Gal Chechik, Naama Friedmann, and Michal Rosen-Zvi. 2015. Probabilistic graphical models of dyslexia. In *Knowledge Discovery and Data Mining (KDD)*, pages 1919–1928, Sydney, Australia.

Made by Dyslexia. 2019. Dyslexia in schools: A survey. http://madebydyslexia.org/assets/downloads/Dyslexia_In_Schools_2019.pdf (2019/9/8).

Ana Marcet, Manuel Perea, Ana Baciero, and Pablo Gómez. 2019. Can letter position encoding be modified by visual perceptual elements? *Quarterly Journal of Experimental Psychology*, 72(6):1344–1353.

Genevieve McArthur, Saskia Kohnen, Linda Larsen, Kristy Jones, Thushara Anandakumar, Erin Banales, and Anne Castles. 2013. Getting to grips with the heterogeneity of developmental dyslexia. *Cognitive neuropsychology*, 30:1–24.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119. Curran Associates, Inc.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, Pennsylvania.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *Workshop on Autodiff, Advances in Neural Information Processing Systems (NIPS))*, Long Beach, California.

R Core Team. 2013. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/.

Luz Rello and Miguel Ballesteros. 2015. Detecting readers with dyslexia using machine learning with eye tracking measures. In *Web for All Conference*, pages 16:1–16:8, Florence, Italy.

Alexander Rush. 2018. The annotated transformer. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 52–60, Melbourne, Australia.

William Shakespeare. 1997. Shakespeare's sonnets. http://www.gutenberg.org/ebooks/1041 (2019/10/24).

Sally E. Shaywitz and Bennett A. Shaywitz. 2005. Dyslexia (specific reading disability). *Biological Psychiatry*, 57(11):1301–1309.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3104–3112. Curran Associates, Inc., Red Hook, NY.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008. Curran Associates, Inc., Red Hook, NY.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.