Language Modeling by Clustering with Word Embeddings for Text Readability Assessment

Miriam Cha
Harvard University
miriamcha@fas.harvard.edu

Youngjune Gwon
Harvard University
gyj@eecs.harvard.edu

H. T. Kung
Harvard University
kung@harvard.edu

ABSTRACT

We present a clustering-based language model using word embeddings for text readability prediction. Presumably, an Euclidean semantic space hypothesis holds true for word embeddings whose training is done by observing word co-occurrences. We argue that clustering with word embeddings in the metric space should yield feature representations in a higher semantic space appropriate for text regression. Also, by representing features in terms of histograms, our approach can naturally address documents of varying lengths. An empirical evaluation using the Common Core Standards corpus reveals that the features formed on our clustering-based language model significantly improve the previously known results for the same corpus in readability prediction. We also evaluate the task of sentence matching based on semantic relatedness using the Wiki-SimpleWiki corpus and find that our features lead to superior matching performance.

CCS CONCEPTS

•**Information systems** → **Clustering;**

KEYWORDS

Readability assessment, clustering-based language model

1 INTRODUCTION

Predicting reading difficulty of a document is an enduring problem in natural language processing (NLP). Approaches based on shallow-length features of text date back to 1940s [9]. Remarkably, they are still being used and extended with more sophisticated techniques today. In this paper, we use word embeddings to compose semantic features that are presumably beneficial for assessing text readability. Encouraged by the recent literature in applying language models for better prediction, we aim to build a clustering-based language model using word vectors learned from corpora. The resulting model is expected to reveal semantics at a higher level than word embeddings and provide discriminative features for text regression.

As pioneering work in text difficulty prediction, Flesch [9] explored on shallow-length features computed by averaging the number of words per sentence and the number of syllables per word. The intent was to capture sentence complexity with the number of words, and word complexity with the number of syllables. Chall [5] claimed the reading difficulty as a linear function of shallow-length features. Kincaid [13] introduced a linear weighting scheme that became the most common measure of reading difficulty based on shallow-length features. More sophisticated algorithms that measure semantics by word frequency counts and syntax from sentence length [22] and language modeling [6] have shown significant performance gains over classical methods.

Modern approaches treat text difficulty prediction as a discriminative task. Schwarm *et al.* [21] presented text regression based on support vector machine (SVM). Peterson *et al.* [20] used both SVM classification and regression for improvement. NLP researchers went beyond the shallow features and looked into learning complex lexical and grammatical features. Flor *et al.* [10] proposed an algorithm that measures lexical complexity from word usage. Vajjala *et al.* [24] formulated semantic and syntactic complexity features from language modeling, which resulted some improvement. Class-based language models, trained on the conditional probability of a word given the classes of its previous words, were commonly used in the literature [3, 23]. Brown clustering [4], a popular class-based language model, can learn hierarchical clusters of words by maximizing the mutual information of word bigrams.

Our text learning is founded on word embeddings. Bengio *et al.* [1] proposed an early neural embedding framework. Mikolov *et al.* [18] introduced the Skip-gram model for efficient training with large unstructured text, and Paragraph Vector [14] and character *n*-grams [2], all of which we use for our implementation in this paper, followed on. Most word embedding algorithms build on the distributional hypothesis [11] that word co-occurrences imply similar meaning and context. Word embeddings span a high-dimensional semantic space where the Euclidean distance between word vectors measures their semantic dissimilarity [12].

Under the Euclidean semantic space hypothesis, we argue that clustering of word vectors should unveil a clustering-based language model. In particular, we propose two clustering methods to construct language models using Brown clustering and K-means with word vectors. Our methods are language-independent and data-driven, and we have empirically validated their superior performance in text readability assessment. Specifically, our experiment on the Common Core Standards corpus reveal that the language model learned by K-means significantly improves readability prediction against contemporary approaches using the lexical and syntactic features. In another experiment with the Wiki-SimpleWiki corpus, we show that our features can correctly identify sentence pairs of the similar meaning but written in different vocabulary and grammatical structure.

For text with easy readability, difference in reading difficulty is resulted from different document length, sentence structure, and

CIKM'17 , Singapore, Singapore

DOI: 10.1145/3132847.3133104

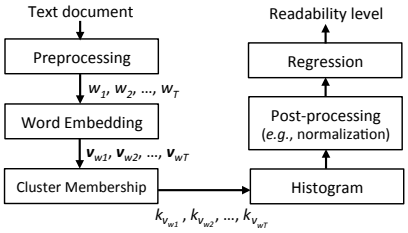

Figure 1: Our system pipeline

word usage. For documents at higher reading levels, however, features with richer linguistic context about domain, grammar, and style are known to be more relevant. For example, based on shallow features, "To be or not to be, that is the question" would likely be considered easier than "I went to the store and bought bread, eggs, and bacon, brought them home, and made a sandwich." Therefore, we need to capture all semantic, lexical, and grammatical features for distinguishing documents at all levels.

We organize the rest of this paper as follows. In Section 2, we describe our approach centered around neural word embedding and probabilistic language modeling. We will explain each component of our approach in detail. Section 3 presents our experimental methodology for evaluation. We will also discuss the empirical results. Section 4 concludes the paper.

2 APPROACH

We review embedding schemes, clustering algorithms, and regression method used in the paper, and describe our overall pipeline.

2.1 Word embeddings

Skip-gram. Mikolov *et al.* [18] proposed the Skip-gram method based on a neural network that maximizes

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0}\log\, p(w_{t+j}|w_t) \tag{1}$$

where the training word sequence $w_1, w_2, \ldots, w_T$ has a length T. With w_t as the center word, c is the training context window. The conditional probability can be computed with the softmax function

$$p(w_{t+j}|w_t) = \frac{e^{s(w_t, w_{t+j})}}{\sum_{w'} e^{s(w_t, w')}} \tag{2}$$

with the scoring function $s(w_t, w_{t+j}) = \mathbf{v}_{w_t}^{\top}\cdot\mathbf{v}_{w_{t+j}}$. The embedding $\mathbf{v}_{w_t}$ is a vector representation of the word w_t.

Bag of character *n*-grams. Bojanowski *et al.* [2] proposed an embedding method by representing each word as the sum of the vector representations of its *character* n-grams. To capture the internal structure of words, a different scoring function is introduced

$$s(w_t, w_{t+j}) = \sum_{g\in G_{w_t}} \mathbf{z}_g^{\top}\cdot\mathbf{v}_{w_{t+j}}. \tag{3}$$

Here, G_{w_t} is the set of n-grams in w_t. A vector representation $\mathbf{z}_g$ is associated to each n-gram g. This approach has an advantage in representing unseen or rare words in corpus. If the training corpus is small, character n-grams can outperform the Skip-gram (of words) approach.

2.2 Paragraph embedding

Distributed bag-of-words. While Skip-gram and character n-grams can embed a word into a high-dimensional vector space, we eventually need to compute a feature vector for the whole document. Le *et al.* [15] introduced Paragraph Vector that learns a fixed-length vector representation for variable-length text such as sentences and paragraphs. The distributed bag-of-words version of Paragraph Vector has the same architecture as the Skip-gram model except that the input word vector is replaced by a paragraph token.

2.3 Clustering

Brown clustering. Brown *et al.* [4] introduced a hierarchical clustering algorithm that maximizes the mutual information of word bigrams. The probability for a set of words $w_1, w_2, \ldots, w_T$ can be written as

$$\prod_{t=1}^{T} p(w_t|C(w_t))\, p(C(w_t)|C(w_{t-1})) \tag{4}$$

where $C(\cdot)$ is a function that maps a word to its class, and $C(w_0)$ is a special start state. Brown clustering hierarchically merges clusters to maximize the quality of C. The quality is maximized when mutual information between all bigram classes are maximized. Although Brown clustering is commonly used, a major drawback is its limitation to learn only bigram statistics.

***K*-means.** Because word embeddings span a semantic space, clusters of word embeddings should give a higher semantic space. We perform K-means on word embeddings. The resulting clusters are word classes grouped in semantic similarity under the Euclidean metric constraint. Given word embeddings $\mathbf{v}_{w_1}, \mathbf{v}_{w_2}, \ldots, \mathbf{v}_{w_T}$ learned from a corpus, we find the cluster membership for a word w_t as

$$k_{\mathbf{v}_{w_t}} = \arg\min_j \|\mathbf{c}^{(j)} - \mathbf{v}_{w_t}\|_2^2 \tag{5}$$

where $\mathbf{c}^{(j)}$ is the jth cluster centroid.

2.4 Regression

We consider linear support vector machine (SVM) regression

$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ s.t. } -\epsilon \le y^{(i)} - (\mathbf{w}\cdot\mathbf{x}^{(i)} + b) \le \epsilon\ \forall i \tag{6}$$

where the regressed estimate $\mathbf{w}^{\top}\cdot\mathbf{x}^{(i)} + b$ for ith input $\mathbf{x}^{(i)}$ is optimized to be bound within an error margin ϵ from the ground-truth label $y^{(i)}$. SVM trains a bias term b to better compensate regression errors along the weight vector $\mathbf{w}$. We train SVM regression using feature vectors formed on word embedding and clustering to predict the readability score.

2.5 Pipeline

Figure 1 depicts our prediction pipeline using word clusters precomputed by K-means on word embeddings. When a document of an unknown readability level arrives, we preprocess tokenized text input and compute word vectors using trained word embeddings. We compute cluster membership on word vectors, followed by average pooling. For cluster membership, we perform the 1-of-K hard assignment for each word in the document. Then we compute the histogram of cluster membership. By representing features in terms of histograms our approach can naturally address

Table 1: Baseline regression results

System	Spearman	Pearson
bag-of-words	0.373	0.433
word2vec	0.579	0.629
fastText	0.670	0.639
doc2vec	0.525	0.539

Table 2: Clustering-based language model results

System	Spearman	Pearson
Brown clustering	0.546 (0.430)	0.534 (0.443)
word2vec +K-means	0.711 (0.670)	0.705 (0.664)
fastText +K-means	0.825 (0.758)	0.822 (0.810)

Table 3: Performance comparison summary

System	Spearman	Pearson
Text length	-	0.36
Flesch-Kincaid	-	0.49
Flor *et al.* [10]	-	-0.44
Lexile [1]	0.50	-
ATOS [2]	0.59	-
DRP [3]	0.53	-
REAP [4]	0.54	-
Reading Maturity [5]	0.69	-
SourceRater [6]	0.75	-
Vajjala *et al.* [24]	0.69	0.61
Our approach	**0.83**	**0.82**

documents of varying lengths. After some post-processing (*e.g.*, unit-normalization), we regress the readability level.

3 EXPERIMENTAL EVALUATION

Following Vajjala *et al.* [24], we evaluate readability level prediction with the Common Core Standards corpus [7] and sentence matching with the Wiki-SimpleWiki corpus [25].

3.1 Common Core Standards Corpus

This corpus of 168 English excerpts are available as the Appendix B of the Common Core Standards reading initiative of the US education system. Each text excerpt is labeled with a level in five grade bands (2-3, 4-5, 6-8, 9-10, 11+) as established by educational experts. Grade levels 2.5, 4.5, 7, 9.5, and 11.5 are used as ground-truth labels. We cut the corpus into `train` and `test` sets in an uniformly random 80-20 split, resulting 136 documents for training and 32 for test.

Evaluation metric. For fair comparison with other work, we adopt Spearman's rank correlation and Pearson correlation computed between the ground-truth label and regressed value.

Preprocessing. We convert all characters to lowercase, strip punctuations, and remove extra whitespace, URLs, currency, numbers, and stopwords using the NLTK Stopwords Corpus [17].

Features. There are two levels of features. At the word-vector level, we perform weighted average pooling of word embeddings to compose per-document feature vector. We have tried tf-idf and uniform weighting schemes. Brown clustering of *words* yields the word-vector level features as well. On the contrary, K-means clustering of *word vectors* yields higher-level features in terms of cluster structures. For Brown and K-means, we replace each word in a document with its numeric cluster ID and compute the histogram of cluster membership as per-document feature vector. For histogram computing, we consider binary (on/off) and traditional bin counts.

Word and paragraph embeddings. We use `word2vec` for the Skip-gram word embeddings. We have first tried out the WIKI and AP-NEWS pretrained `word2vec` models. Eventually, we use TensorFlow to train `word2vec` model from the Common Core Standards corpus. We have optimized the word-vector dimension hyperparameter between 32 and 300.

We use `fastText` for character n-gram word embeddings. Similar to our `word2vec` experiment, we have tried the WIKI and AP-NEWS pretrained models for `fastText` before training our own. While training, we use the negative sampling loss function with word-vector dimensions 32 to 300 and context window size of 5.

We use `doc2vec` that implements Paragraph Vector. We have not trained our own `doc2vec` model and opted for the WIKI and AP-NEWS pretrained `doc2vec` models.

Brown clustering. We use an open-source implementation by Liang *et al.* [16]. We have fine-tuned the number of cluster hyperparameter by varying between 10 and 200.

***K*-means clustering.** After embedding all words in each document, we run K-means. We fine-tune K within 10 to 200.

SVM regression. We use LIBLINEAR [8] for SVM regression and configure as the ℓ_2-regularized ℓ_2-loss linear solver with unit bias. The SVM complexity hyperparameter is optimized between 10^{-5} and 1. Our choice of linear SVM is made after also trying out a single-layer perceptron neural network regression with the number of neurons in 0.1x to 1x the feature vector dimension.

Results and discussion. Our baseline results with pretrained models are shown in Table 1. `Bag-of-words` performs poorly, and `word2vec` performs better than `doc2vec`. We suspect that the benefit of `doc2vec` is not realized on this corpus due to its limited length. We find `fastText` superior over `word2vec` and `doc2vec`. Pretrained WIKI outperforms AP-NEWS. We only report WIKI results.

Table 2 presents results on clustering-based language models: Brown clustering on words and K-means on trained word vectors using the corpus. Presented correlation values are for binary (inside parenthesis) and traditional bin counts. While binary counters could be robust against ambiguities resulting from repeated texts in a document, this advantage is not present in the corpus we use here. Brown clustering on words has similar performance to baseline embedding schemes. The comparable performances are expected, because both Brown clustering and the baseline embedding schemes are performed on the raw words. We can improve performance further with K-means clustering on word vectors. Rather than training word vector models on WIKI, training with the Common Core Standards corpus improves the correlation. `fastText` with K-means works the best.

Table 3 presents a summary that compares performances of our approach and the previous work. Flor *et al.* [10] implemented prediction scheme based on lexical tightness and compared their method against baselines such as text length and Flesch-Kincaid [13] in Pearson correlation. Nelson *et al.* [19] wrote a summary of

[1] http://lexile.com
[2] http://www.renaissance.com
[3] http://questarai.com
[4] http://reap.cs.cmu.edu
[5] http://readingmaturity.com
[6] http://naeptba.ets.org

Table 4: Average probability P_N that a Wiki sentence and its SimpleWiki counterpart are within the Nth nearest neighbors in the semantic feature vector space

	N			
	1	2	3	4
P_N	0.926	0.947	0.955	0.959

commercial softwares' performances in Spearman correlation. Most recently, Vajjala *et al.* [24] implemented a scheme that uses lexical, syntactic, and psycholinguistic features. Our highest correlation for Spearman is 0.83, and 0.82 for Pearson, both of which are better than the best case reported by the previous work.

3.2 Wiki-SimpleWiki Corpus

We demonstrate our features derived from clustering of word embeddings are effective in another application concerning sentence matching. The corpus for this application consists of 108,016 aligned sentence pairs of the same meaning drawn from (ordinary) Wikipedia and Simple Wikipedia.[7] Simple Wikipedia uses basic vocabulary and less complex grammar to make the content of Wikipedia accessible to audiences of all reading skills.

Task and metric. We evaluate whether or not the feature vector for an ordinary sentence formed by the proposed feature scheme can correctly predict its counterpart sentence. We sample 1,000 sentence pairs. Among all 1,000 pairs, we compute the probability P_N that ordinary sentences and their simple counterparts are N nearest neighbors in the semantic space. We vary $N = 1$ to 4.

Features. We use our best feature scheme, word embedding by `fastText` and K-means, found in Section 3.1. To compute sentence embedding, we average-pool all word embeddings in the sentence.

Results and discussion. As Table 4 shows, using only the nearest neighbor, we already achieve $P_N = 0.959$; as N grows, we can contain different sentences of the same meaning with probability approaching 1. This implies that despite differences in grammatical structure and word usage, when underlying semantics are shared between two sentences, they are mapped closely each other in the feature space.

4 CONCLUSION

Word vectors learned on neural embedding exhibit linguistic regularities and patterns explicitly. In this paper, we have introduced a regression framework on clustering-based language model using word embeddings for automatic text readability prediction. Our experiments with the Common Core Standards corpus demonstrate that features derived by clustering word embeddings are superior to classical shallow-length, bag-of-words, and other advanced features previously attempted on the corpus. We have further evaluated our approach on sentence matching using the Wiki-SimpleWiki corpus and showed that our method can effectively capture semantics even when sentences are written with different vocabulary and grammatical structures. For future work, we plan to continue our experiments with more diverse languages and larger datasets.

[7] http://simple.wikipedia.org

5 ACKNOWLEDGMENTS

This work is supported by the MIT Lincoln Laboratory Lincoln Scholars Program and in part by gifts from the Intel Corporation and the Naval Supply Systems Command award under the Naval Postgraduate School Agreements No. N00244-15-0050 and No. N00244-16-1-0018.

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *JMLR* 3 (2003), 1137–1155.
[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).
[3] Jan A Botha and Phil Blunsom. 2014. Compositional Morphology for Word Representations and Language Modelling.. In *ICML*. 1899–1907.
[4] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based *n*-gram Models of Natural Language. *Computational linguistics* 18, 4 (1992), 467–479.
[5] J.S. Chall. 1958. *Readability: An Appraisal of Research and Application.* Ohio State U. Press.
[6] Kevyn Collins-Thompson and Jamie Callan. 2005. Predicting Reading Difficulty with Statistical Language Models. *Journal of Association for Information Science and Technology* 56, 13 (Nov. 2005), 1448–1462.
[7] Council of Chief State School Officers. 2010. Common Core State Standards for English Language Arts & Literacy in History/Social Studies, Science, and Technical Subjects. Appendix B: Text Exemplars and Sample Performance Tasks. http://www.corestandards.org/assets/Appendix_B.pdf. (2010).
[8] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *JMLR* 9 (2008), 1871–1874.
[9] Rudolph Flesch. 1948. New Readability Yardstick. *Journal of Applied Psychology* 32, 3 (June 1948), 221–233.
[10] Michael Flor, Beata Beigman Klebanov, and Kathleen M Sheehan. 2013. Lexical tightness and text complexity. In *Proceedings of the 2th Workshop of Natural Language Processing for Improving Textual Accessibility (NLP4ITA)*. Citeseer, 29–38.
[11] Zellig S Harris. 1954. Distributional structure. *Word* 10, 2-3 (1954), 146–162.
[12] Tatsunori B Hashimoto, David Alvarez-Melis, and Tommi S Jaakkola. 2016. Word embeddings as metric recovery in semantic spaces. *Transactions of the Association for Computational Linguistics* 4 (2016), 273–286.
[13] J.P. Kincaid. 1975. *Derivation of New Readability Formulas: (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel.* U.S. Naval Air Station, Memphis.
[14] Jey Han Lau and Timothy Baldwin. 2016. An Empirical Evaluation of Doc2vec with Practical Insights into Document Embedding Generation. *arXiv preprint arXiv:1607.05368* (2016).
[15] Quoc V Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents.. In *ICML*, Vol. 14. 1188–1196.
[16] Liang, P. 2012. Implementation of Brown Hierarchical Word Clustering Algorithm (v1.3). https://github.com/percyliang/brown-cluster. (2012).
[17] Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*. Association for Computational Linguistics, 63–70.
[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).
[19] Jessica Nelson, Charles Perfetti, David Liben, and Meredith Liben. 2012. Measures of text difficulty: Testing their predictive value for grade levels and student performance. *Council of Chief State School Officers, Washington, DC* (2012).
[20] Sarah E. Petersen and Mari Ostendorf. 2009. A Machine Learning Approach to Reading Level Assessment. *Computer Speech and Language* 23, 1 (2009), 89–106.
[21] Sarah E. Schwarm and Mari Ostendorf. 2005. Reading Level Assessment Using Support Vector Machines and Statistical Language Models. In *ACL*.
[22] A.J. Stenner. 1996. Measuring Reading Comprehension with the Lexile Framework. In *North American Conference on Adolescent/Adult Literacy*.
[23] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 384–394.
[24] Sowmya Vajjala and Detmar Meurers. 2014. Assessing the relative reading level of sentence pairs for text simplification.. In *EACL*. 288–297.
[25] Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd international conference on computational linguistics*. Association for Computational Linguistics, 1353–1361.