Machine Learning and Evaluating Technical Documentation

Muchang Bahng, Alejandra Camargo, Nikita Daga
Grant Glass, Christopher Ritter

Overview

NetApp is a technology company that specializes in data storage-retrieval hardware and software systems. It maintains a repository of 65,446 technical documents, in order to provide customers with resources and instructions to efficiently use its products.

We are interested in examining how effective NetApp's product specific documentation is in communicating technical specifications. To ensure that the documents are easily comprehensible, the company aims to build a machine learning model that can generate readability scores for each document in the corpus.

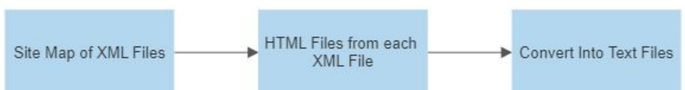

Fig. 1: Data extraction pipeline for all documents.

Methods

NetApp provides an XML sitemap containing HTML links corresponding to each technical document for one of the company's products or services. All files were scraped, parsed, and stored in .txt files using Python packages Requests and BeautifulSoup4. The following analysis methods were employed on this corpus:

- *Dimensionality reduction* of readability scores using linear methods, e.g. principal component analysis (PCA), and nonlinear methods, e.g. uniform manifold approximation and projection (UMAP).
- *Sparsity measure* of the documents, represented as a vector encoding the conditional distribution of an n-gram model.
- *Deep learning* methods to predict a readability score with document embeddings in a moderately high-dimensional vector space using labels generated from UMAP.

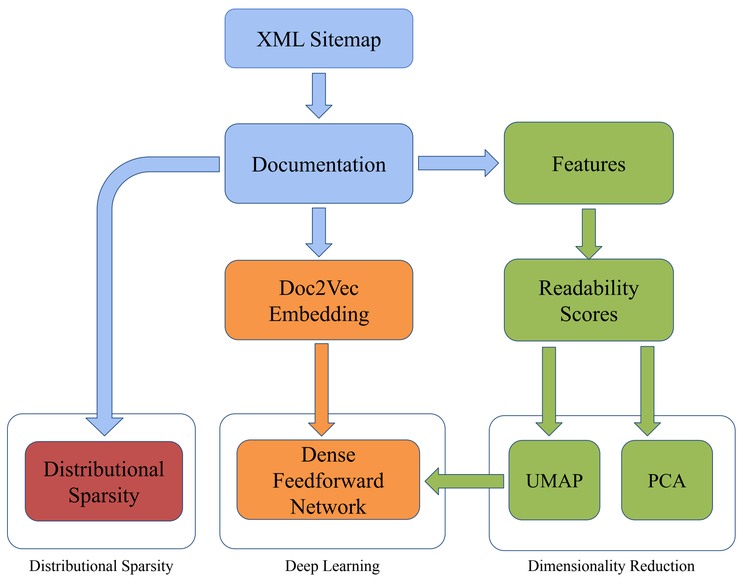

Fig. 2: Workflow diagram of our study.

The relevant features that are used to calculate the readability scores are documented below:

	Words	Sentences	Characters	Syllables	3+ Syllable Words	< 2–3 Syllable Words	< 6–7 Character Words
Anderson's Readability Index (RIX)		×					×
Automated Readability Index (ARI)	×	×	×				
Coleman–Liau	×	×				×	
Danielson–Bryan		×	×				
Dickes–Steiwer Handformel	×	×	×				
Fang's Easy Listening Formula		×				×	
Farr–Jenkins–Paterson	×	×				×	
Flesch	×	×		×			
Flesch–Kincaid	×	×		×			
FORCAST	×					×	
Fucks' Stilcharakteristik	×	×	×				
Gunning Frequency of Gobbledygook (FOG)	×	×			×		
Kuntzsch's Text–Redundanz–Index						×	
Linsear–Write	×				×	×	
LIX Score	×	×					×
Neue Wiener Sachtextformeln (nWS)	×				×		×
Simple Measure of Gobbledygook (SMOG)		×			×		
Strain Index		×		×			
Wheeler–Smith	×	×				×	

Fig. 3: Nineteen metrics commonly used to assess the readability of documents, with their corresponding definitions. Note: Kuntzsch's Text-Redundanz-Index (KTRI), which accounts for the number of foreign words in the document, was not employed in our analysis of the English-language corpus.

Dimensionality Reduction

A total of 18 readability scores were extracted from the 65,446 documents, represented as vectors in $\mathbb{R}^{18}$. The 18 readability scores computed for each document contained a lot of overlapping information as seen above. We implemented PCA and UMAP to capture the principal components of the data and identify nonlinear structure within the dataset.

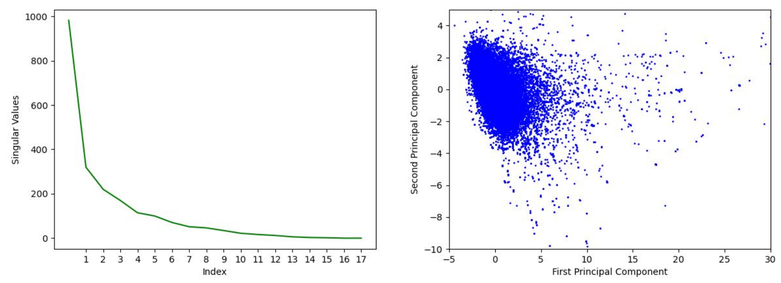

Fig. 4: Singular values of the dataset design matrix (left), with explained variance of 81.78% and 8.69% in first and second principal components, respectively. Scatter plot of dimensionality-reduced vectors in $\mathbb{R}^2$.

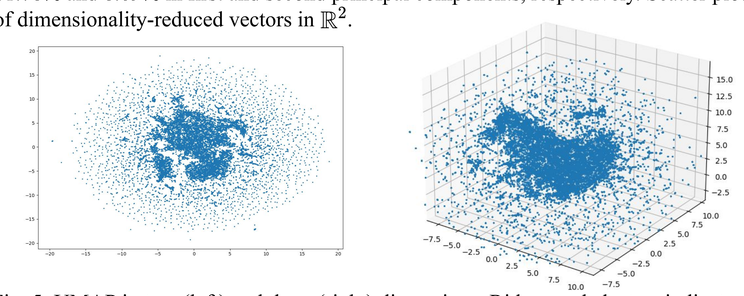

Fig. 5: UMAP in two (left) and three (right) dimensions. Ridges and clusters indicate that there are nonlinear structures in the data.

Distributional Sparsity

Given a set of n words $w_1, w_2, \ldots, w_n$, the conditional distribution of the next word $\mathbb{P}(w_{n+1} \mid w_1, \ldots, w_n)$ can be generated by the n-gram model which stores a dictionary of n-grams from the corpus.

```
...
('that', 'help'): ['you'],
('that', 'is'): ['packaged'],
('the', 'active'): ['iq'],
('the', 'api'): ['documentation', 'documentation', 'calls'],
('the', 'apis'): ['are', 'that', 'under', 'also', 'used', 'under'],
('the', 'applicable'): ['api'],
...
```

A greater number of (distinct) possible next words w_{n+1} implies higher entropy, indicating more possible words to choose from. This correlates with less predictability of future words, indicating lower readability. We convert each document to a vector storing the counts of possible future words.

$$\mathbf{v}_{\text{doc}} = (v_1, \ldots, v_d), \text{ where } v_j = \#\text{ of possible next words}$$

We measure the sparsity of this vector using the PQ index (which is permutation-invariant) defined

$$I_{p,\,q}(w) = 1 - d^{\frac{1}{q}-\frac{1}{p}} \frac{||w||_p}{||w||_q}, \quad ||w||_p = \left(\sum_{i=1}^{d} |w_i|^p \right)^{1/p}$$

A higher index indicates higher sparsity, indicating lower readability.

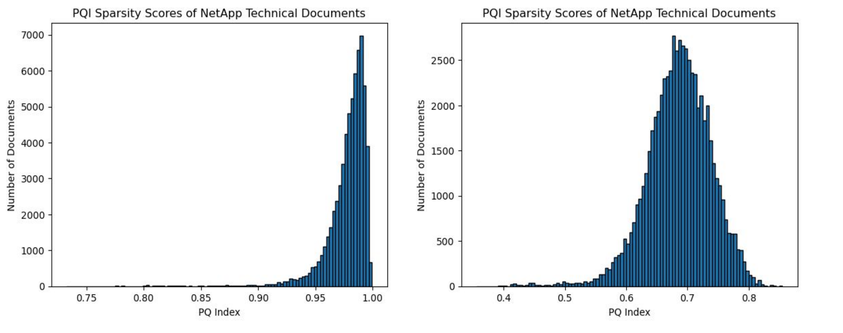

Fig. 6: PQI sparsity scores on 2-gram vectors of NetApp technical documentation with p=2, q=1 (left) and p=3, q=2 (right).

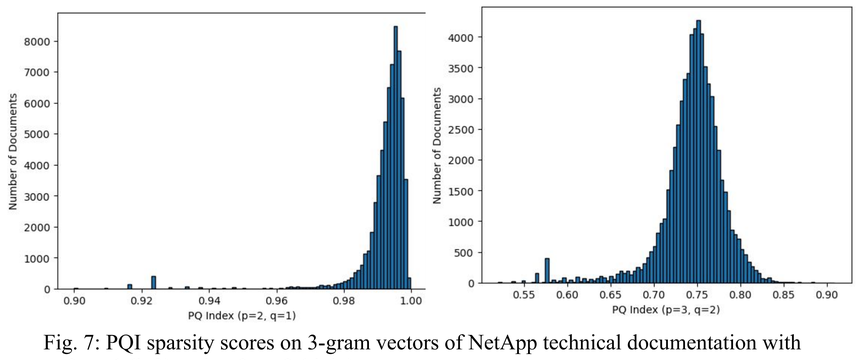

Fig. 7: PQI sparsity scores on 3-gram vectors of NetApp technical documentation with p=2, q=1 (left) and p=3, q=2 (right).

Neural Network w/ Embeddings

To capture further nonlinearities, we train a feedforward neural net on the document data. Our samples consist of doc2vec embeddings of each document with 1-dimensional labels generated from UMAP, which essentially reduces the 18 readability scores to a single scalar.

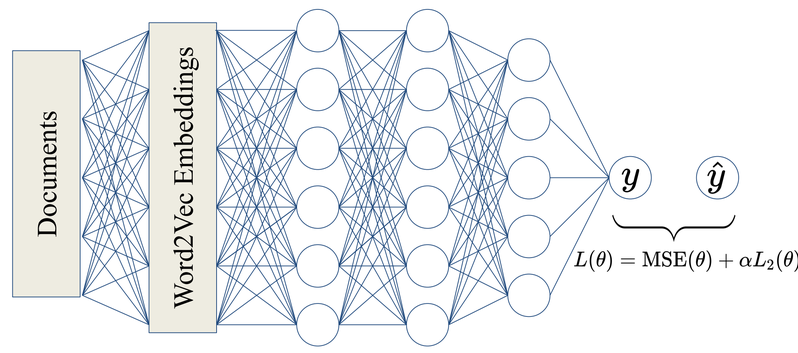

Fig. 8: Deep learning architecture (3-layer multilayer perceptron w/ 300 parameters per layer and leaky RELU activation) for learning readability scores of technical documentation with UMAP labels. Trained using Adam optimizer with dropout (50%) and L2 regularization ($\alpha = 0.1$).

The training and testing loss deviates around certain values rather than showing consistent improvement, even with varying step sizes, optimizers, regularization parameters, and neural net size. This indicates that the document and UMAP embeddings may be uncorrelated, leading to difficulties of the neural net finding patterns. Another potential problem may be that the loss landscape may be too "jagged," making it harder for Adam to find local minima.

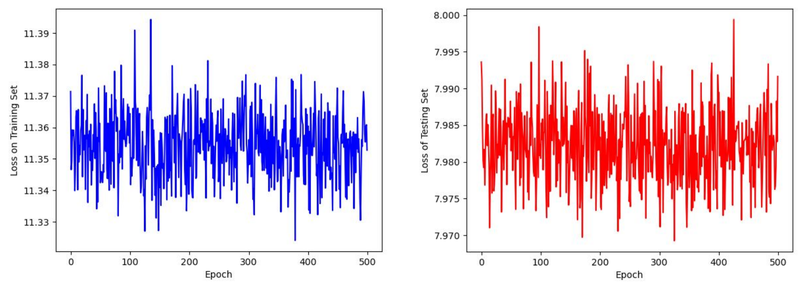

Fig. 9: Training loss (left) and testing loss (right) of the neural net trained over 500 epochs with Adam optimizer of batch size 1024. The training and testing losses seem to deviate around 11.356 and 7.984, respectively.

Conclusion

The results indicate that utilizing a combination of established readability scores is useful in detecting readability of technical documentation. We are able to identify clusters and correlations between the readability features of the NetApp corpus, and further work can be done on improving the interpretability of these algorithms.

Measuring the distributional sparsity of words in the document, which is directly related to the entropy of the conditional distribution of future words, also provides insights on how the readability is distributed across the entire corpus. We found that these distributions are left-skewed, indicating that few documents are extremely easy to read, while most of the documents are either clustered towards high sparsity values near 1 or roughly follow a Gaussian distribution.

Finally, we have provided a deep learning architecture for measuring readability, which can be used in the future with higher-quality datasets containing more accurate labels on readability.

References

- Benoit, K. (n.d.). *Calculate readability*. Quanteda. https://quanteda.io/reference/textstat_readability.html
- E. Diao, G. Wang, J. Zhang, Y. Yang, J. Ding, and V. Tarokh. Pruning Deep Neural Networks from a Sparsity Perspective. *ICLR 2023*
- Q. Le and T. Mikolov. 2014. Distributed Representations of Sentences and Documents. Proceedings of the 31st International Conference on Machine Learning (PLMR) 32(2): 1188-1196.