TextCompare

Prof. Tony Eng, Antonio Rivera, Vahid Fazel-Rezai

1 Introduction

Learning a new language can be daunting because it can be hard to know where to start. There are lots of words in a given language, and one has to somehow organize them in some order to learning, first the basic words first and slowly more and more advanced words. One way to acquire new words is to read books or watch movies in the language. We propose a system hinged around comparing pairs of texts to map out a reading order for a language learner.

On a high level, our approach is to model a set of text documents (e.g. book or movie transcripts) as a complete graph with edges indicating differences in skill level, using which we can find a path from the current skill level to the desired skill level, with conditions on the number of edges and distance of each step. Start and end skill levels can be represented as a list of words and included as nodes in the graph. An extension of this system is to work on a word-by-word level, potentially discerning which words are more basic or more advanced. The three essential questions that must be answered are:

- What properties of a piece of text are we trying to measure?
- What metric(s) (i.e. formulas on document word frequencies) capture these properties?
- Given all pairwise distances of a set of books, as well as a start what set of conditions

In addition, prior to answering the above questions we must decide on and acquire toy data to prototype on, and after answering these questions we need to address the problem of how to make the system practical and usable.

2 Data

We began by collecting data of the following types:

- Control documents with specific properites (generated manually, e.g. single repeated word).
- Children's books.
- Nursery rhymes.

These were chosen because they are short and simple, allowing us to reason on a low level and manually compare documents. Also, children's books are primarily used to develop language, so we can maybe incorporate subjective or anecdotal insights.

3 Document Properties

Our goal is to use comparisons between documents to infer their relative difficulties. In our case, difficulty includes two factors:

- Size of vocabulary.
- How advanced the vocabulary is.

The distinction between how advanced a vocabulary is is important–consider several advanced books on the same topic and another medium book on a different topic.

4 Similarity and Distance Metrics

Consider two documents X and Y.

$$\begin{aligned}
a &= \text{Number of unique words only in A}\\
b &= \text{Number of unique words only in B}\\
c &= \text{Number of unique words in common}\\
a &= \text{Number of only in A}\\
b &= \text{Number of only in B}\\
c &= \text{Number of in common}
\end{aligned}$$

The metrics used to compare two texts can be split into three groups:

1. Metrics that use only c (i.e. overlap comparison). These include Jaccard index, Canberra, Sorenson, and Minkowski2.
2. Metrics that use a, b, and c. Examples of these are TF, TF-IDF, and their variants, such as sublinear TF-IDF and TF-ITTF.
3. Asymmetric metrics. The main one used is Tversky Index, but one could also consider simpler metrics such as number of new words or number of new occurrences.

TF-IDF and Variants:

These metrics measure the angle between word vector representations of texts. The difference between the variants is how those word vectors are determined. The simplest form is TF (Term Frequency). This simply constructs word vectors where the ith value in the vector denotes the frequency of term i for that particular text. Given two texts A and B with word vectors X and Y respectively, we then simply calculate the angle between the texts:

$$\arccos\left(\frac{X \cdot Y}{\|X\|\|Y\|}\right)$$

Asymmetric Metrics:

Tversky index is defined, for sets of words X and Y, as:

$$\frac{|X \cap Y|}{|X \cap Y| + \alpha|X - Y| + \beta|Y - X|}$$

with $\alpha, \beta \geq 0$. It should be noted that if $\alpha = \beta$, then the metric is symmetric. For our purposes, I think that setting $\alpha = 0$ will give us the metric we want. When considering whether a child can read book Y after reading book X, what we care about is how many new words there are in Y. Thus, letting $\alpha = 0$ and $\beta \neq 0$ reflects this. This intuition has been reflected when attempting to maximize/minimize particular distances in a test set of books.

Distance:

- Vocabulary (number of new words)
-

Similarity:

- Vocabulary (number of overlapping words)

5 Path Finding

In this section we assume we are given a set of documents with one or more distance metrics that can be evaluated on each pair of documents. We model this as one or more complete graphs where nodes are documents and edges may be directed. There may even be directed cycles that we must deal with.

Possible conditions on a path could be:

- Minimize number of steps in path.
- Minimize largest distance across all steps in path.
- Maximize total number of new words.

5.1 Ordering

In this use case, we are given a set of documents and simply need to sort the in order of least to most advanced. Mathematically, we need to find a path that hits every node.

5.2 Optimizing start to finish

In this case, there are start and end nodes included in the graph(s). (These can be documents or just lists of words treated like a document.) We wish to find the best path from start to finish for some definition of best.