Kevin Wang
https://umap-learn.readthedocs.io/en/latest/document_embedding.html

Explanation to *Document Embedding with UMAP*

Document embedding using UMAP:

1. ***Libraries:*** The tutorial begins by importing relevant libraries pandas, UMAP, and matplotlib. Pandas is designed to make working with relational or labeled data easier through its expressive data structures. Additionally, it can be used for data manipulation operations for data cleaning, sorting, filling, and plotting. UMAP is a dimension reduction technique similar to t-SNE but can also be applied for general non-linear dimension reduction. Matplotlib is used to create visualizations in python.
2. ***Fetching Data and Data Frame creation:*** Next, The 20 newsgroup text dataset is downloaded. A data frame containing the target labels is created to be used in plotting and visualization
3. ***Raw Counts:*** The bag-of-word approach is used to create a word document matrix. The bag-of-word model is a model of text which disregards word order. Usually, this is done as word frequencies can be normalized with the inverse document frequency. The word document matrix stores the frequency of each word within each document. The CountVecotorizer is used for this purpose, with preprocessing steps such as tokenization, removal of stopwords, and filtering of words occurring less than 5 times in the entire corpus
4. ***Reducing Dimensionality:*** Dimensionality is the number of features used to represent each document in the corpus. In this case, it represents the number of unique words across all documents. Reducing dimensionality is ideal, as high dimensionality can lead to the exponential growth of the volume of data space, leading to computational complexity. Additionally, low dimensional graphs are much easier to look at.
    a. ***UMAP:*** While PCA is one of the most popular linear dimension reductions, it only works well when the first 2 principal components account for most of the variation in the data. This is because PCA only plots the 2 highest PCs, while ignoring the rest. UMAP is an alternative without this flaw, as it constructs a graph representation of the data, connecting points based on their proximity within the high-dimensional space.
        i. Selecting a random point that we will call point A for now, UMAP calculates the distance between point A and every other point within the high-dimensional space using *Hellinger Distance*. Then, UMAP plots all of these distances relative to point A along the x-axis of a graph, with point A set at the origin. Next, the function $\ln_2$(number of high dimensional neighbors) is calculated, with a common default value of the number of high dimensional neighbors being 15. For the sake of simplicity, we will refer to this number as value z. Next, a curve representing the similarity score is graphed with the equation:
        $\mathbf{y = e^{-[(\text{raw dist. - dist. from closest point })/\sigma]}}$
        where raw dist. represents the distance between point A and any other point while dist. from closest point represents the distance between point A and its closest point. The curve that will be graphed is shaped in such a way that the

y-axis of the nearest neighbors of point A will add up to value z. However, as distance is fixed, to create a graph that fits with z, σ is changed to some arbitrary value. Thus, the y value associated for each point on the x-axis represents its similarity score. It is important to note that the similarity score between two points are not symmetrical, as both of their curves are different. To solve for this UMAP uses the fuzzy union operation:

Symmetrical score = $(S_1 + S_2) - S_1 * S_2$

where S_1 is the similarity score for some point B relative to another point C while S_2 is the similarity score for point C relative to point B.

ii. Given the similarity scores for every point relative to every other point, UMAP uses spectral embedding to initialize a low-dimensional graph.

iii. However, this low-dimensional graph is not very accurate. To solve this, UMAP calculates the low-dimensional similarity scores using a fixed symmetrical t-distribution curve.

Low D. Score = $1/(1 + \alpha d^{2\beta})$

where d represents the distance between two points while α and β represent user defined parameters for the spread and distance between the low-dimensional points. By default, $\alpha = 1.577$ and $\beta = 0.8951$. First, UMAP selects two low-dimensional points to move closer to each other by randomly selecting a pair of points in a neighborhood proportional to their high-dimensional score. We will call one of these points point A again. UMAP calculates the low-dimensional score between these two points based on their distances, a value we can call "neighbor score". After that, UMAP selects a point that point A should move away from by randomly selecting a point outside of point A's neighborhood. We can call this "not neighbor score". Given the "neighbor" and "not neighbor score", UMAP uses the equation:

Cost = log(1/neighbor) + log[1/(1 - not neighbor)]

Next UMAP graphs this cost function with the position of point A on the x axis and the cost associated with that location on the y axis. Through finding the derivative of this graph, it allows UMAP to use Stochastic gradient descent to optimize the position of point A. This is the optimized position of A on the low-dimensional graph. The same process is done for every other point to create a low-dimensional graph reflective of the high-dimensional graph

b. ***Hellinger Distance:*** Hellinger Distance is usually used to quantify the difference between two probability distributions. However, in this case it quantifies the distance between two high-dimensional points. Unfortunately, I'm not sure how Hellinger Distance was adapted for this, although it is possible as it was mentioned in UMAP's 0.4 release notes. It may be in here: https://arxiv.org/pdf/2103.11773.pdf

5. ***Tf-idf:*** In addition to all of this, the tutorial also uses tf-idf which is a statistical measure that evaluates how relevant a word is to a document. By multiplying both the number of times a word appears in a document and the inverse document frequency of the word across a corpus.

6. ***Plotting:*** Given the 2-dimensional graph, it is plotted in order to see common words among different groups