6.806- Advanced Natural Language Processing
Massachusetts Institute of Technology December 14, 2015

Sifter, a New Machine Learning Application for Clustering Medical Research Findings

Name: Winter Guerra

Collaborators: None

Code Repository:
`https://github.com/Winter-Guerra/6.806_nlp_cancer_research`

Dataset Location:
`http://nlp-dataset-6806-2015.s3.amazonaws.com/index.html`

1 Abstract

The quantity of medical and scientific literature available to the average scientist is increasing at a rapid pace. However, there is currently no good method for easily extracting information from this multitude of data without extensive human interaction. As a result of this inability to easily sift through data, many important findings from cutting edge medical research go unnoticed by the rest of the scientific community. What is needed is a new tool to simplify the act of organizing medical research data based on clusters of findings and topics. This is what my project, Sifter, aims to do.

Utilizing Amazon Web Services's powerful backend, Sifter cross-references NIH's PubMed Open Access dataset of 1,156,698 full-text XML medical research papers and 82,448 meta-research articles to automatically create training clusters of article topics without human interaction. However, after much trail and error testing multiple feed-forward neural network designs on the data that Sifter created, we were unable to perform better than a random data baseline after 75 epochs. Nonetheless, we believe that this result is due to issues with our implementation of our neural network models and could be remedied in the future.

To assist in advancing this goal, we have published the dataset that Sifter created [here] for the convenience of all researchers whom wish to explore the applications of the Sifter dataset to clustering scientific articles.

2 Introduction

2.1 The Problem

The quantity of medical and scientific literature available to the average scientist is increasing at a rapid pace. However, there is currently no good method for easily extracting information from

this multitude of data without extensive human interaction. As a result of this inability to easily sift through data, many important findings from cutting edge medical research go unnoticed by the rest of the scientific community.

Making this issue worse, scientific articles in large databases such as PubMed are not grouped with respect to the details of their *scientific findings,* but rather are commonly grouped by cross-document similarity, bibliographical-coupling, or other metrics that do not explicitly seek to prioritize groupings of articles based on the intricacies of their scientific findings. For example, given two articles a_1, a_2, bibliographic-coupling may indicate that these articles are related in nature; however, an expert analysis of these articles would indicate that the scientific findings of these two articles are quite different since they analyzed different populations (i.e. male vs. female).

Because of the complexity that factors into analyzing scientific article similarity, labeling of training data for article similarity datasets is commonly done by hand. However, this is very expensive and time-consuming. Thus, this article introduces an experimental automated method of aggregating article similarity data based on *scientific finding* distance metrics for use in higher-level NLP article similarity classifiers.

2.2 Approach

2.2.1 Creating the Training Dataset

To create our ground-truth dataset of article similarity metrics, we performed the following steps.

- We downloaded the NIH Open Access Subset to AWS EBS storage.
- We extracted document-type metadata from all 1,156,698 articles in the NIH dataset to get an in-memory index of all relevant and irrelevant articles.
- Then, we extracted 41 million citations from all meta-research articles in addition to their in-text citation locations.
- We then pruned out all citations that linked to articles outside of our dataset.
- We then created a similarity distance matrix from all remaining citations.
- We then saved the similarity matrix in the form of a sparse distributed hashtable in the Redis in-memory cache for fast read access.

Afterwards, to create the actual DMLP training matrix X that held 3,224,369 concatenated pairs of summary sentence embeddings and their associated relevance (in matrix Y), we performed the following operations.

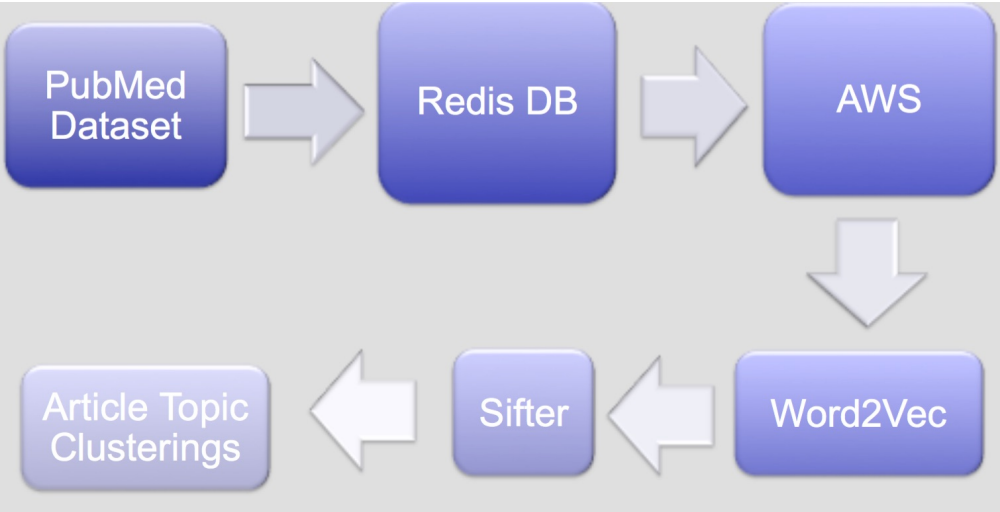

Figure 1: Flowchart of data flow in Sifter.

- We first split our dataset into train/test into (0.75/0.25) subsets by randomly selecting 25% of the article IDs from our available pool of cited research articles. These articles were rigorously held out from dataset generation by blocking all attempts to access these articles as *"article not found."* Thus, this test set served to simulate the addition of new scientific research articles into the PubMed database.

- Then, using the training subset of article IDs, we permuted our data links to create a list of all valid similarities (i.e. `[(text_1, text_2), (text_2, text_1), (text_1, text_345)...  ]`). *Note that this training subset does not contain any articles from the held-out set.*

- To create negative examples for the DMLP, we created another list of pairings that, for each positive pairing (a_1, a_2), created a negative pairing (a_1, a_{bad}) where a_{bad} was an article ID that had no relation to a_1. Due to this construction method, this negative reinforcement list had approximately the same number of links as the positive reinforcement link list. *Note that this training subset does not contain any articles from the held-out set.*

- Finally, we combined both the positive and negative reinforcement list, shuffled the ordering of their example pairings, then converted all document IDs to their summary vector representation. This formed our X_train matrix, a `3.8GB` matrix of size $(2\,530\,435, 400)$.

Lastly, we needed to create our test matrix X_{test} from our held out set of "new, unseen" articles. To do this, we followed much of the same steps used to create X_{train} above. However, there were some small differences.

- For the positive and negative example pair lists, we only chose pairs of examples where both articles were in the held out set. This helped to verify that the classifier had learned well

and was adequately able to create similarity metrics between utterly unseen article summary embeddings.

This method created the `1GB` X_{test} testing matrix of shape $(693\,934, 400)$ that (along with the X_{train} dataset), both reside at our dataset distribution endpoint [here].

3 Experiment

Using the citation distance data harvested from PubMed, we trained a Deep Multilayer Perceptron neural network that takes in a concatenated bag-of-words vector embedding of the summary sentence of 2 articles, then outputs a binary classification of whether the two articles are similar or not (see figure 1).

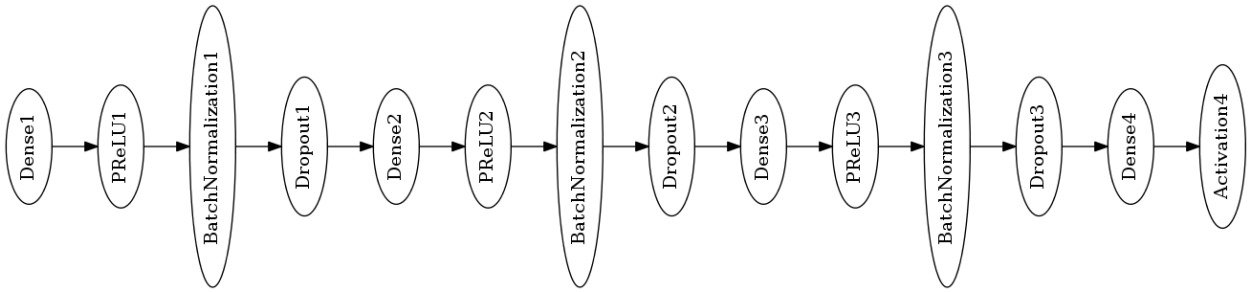

Figure 2: The DMLP network architecture tested in this writeup for Sifter.

Using this simple DMLP model, we were able to achieve the results shown in Table 1 on the held-out test dataset we outlined in part 2.2.1 of this paper.

Table 1: Sifter DMLP Neural Network Accuracy vs Random Choice Baselines

Method	Accuracy
Sifter DMLP Network with 75 training epochs	0.500
Random +/- choice on training set	0.500
Random +/- choice on test set	0.500
Total ratio of +/- labels in test & training set	0.489

4 Conclusions and Next Steps

As we can see from our results in table 1, there is a lot of room for improvement regarding Sifter's neural network implementation. However, we already have multiple concrete ideas for how we can improve Sifter's performance.

First, instead of using a DMLP model that uses a vectorized bag-of-words document summary for evaluating document topic similarity, we would like to revamp our model by using a more complex, sequence-conscious Siamese Convolutional Neural Network (see figure 3). This type of model has already been proven to work well for calculating similarity metrics between image inputs (in the form of the MNIST Digit Dataset), but has also seen some use for evaluating sentence similarities.

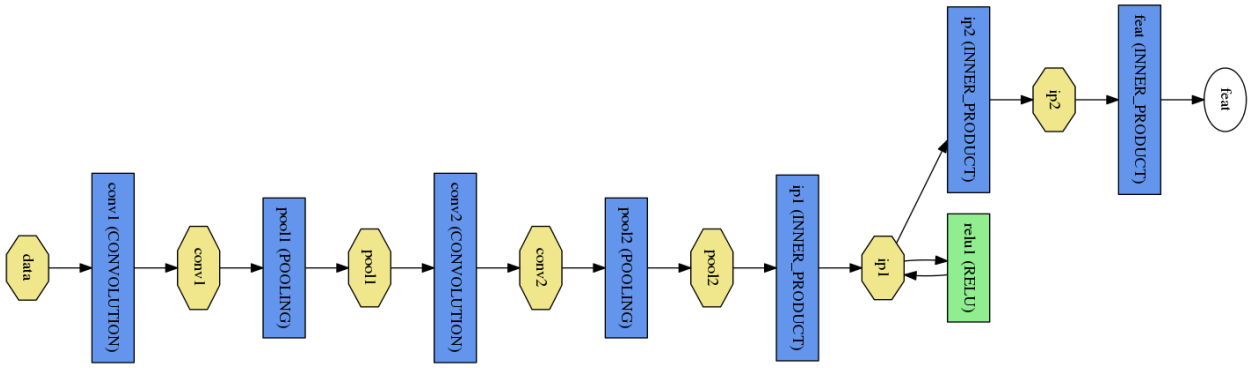

Figure 3: The convolutional Siamese neural network model that we plan to test in the future against Sifter's training data.

In fact, we have already started working on reshaping the Sifter dataset for input into the Siamese Convolutional Neural Network seen in figure 3 in [this github repo.] To do this, we created code that reworked the dataset by converting the document summary representation of each scientific article from 1D vectors of shape $(1, \texttt{embedding\_vector\_dimension})$ to a 2D matrix of sequenced word embeddings. When these 2D vectors were combined together, we got a training matrix of the following shape:

$$(N, \texttt{sentence\_length}, 2 * \texttt{embedding\_vector\_dimension}) \simeq (3\,224\,369, 10, 400)$$
$$\simeq \text{a matrix } \mathbf{51.59GB} \text{ in size}$$

As one can see, at **51.59GB**, this new convolutional training matrix is prohibitively large and creates a whole host of logistical issues regarding its storage, RAM usgae, and computational runtime. This host of logistical issues prevented us from successfully running this dataset against the Siamese Neural Network in figure 3 – even on the largest GPU-enabled cloud computing cluster available on AWS (the `g2.8xlarge` cluster). However, with further research into larger-scale computing techniques, it should be possible to create this dataset and run against our proposed Siamese Neural Network for increased clustering accuracy.