Classification of Invalid Page Content for Topic Modeling Pipelines of Web Browsing Data

Natalie Cygan
Department of Computer Science
Stanford University
cygann@stanford.edu

Abstract

With the great number of articles and resources accessed online, a significant portion of one's intellectual exploration is recorded in the form of browsing history. However, the tabular nature of this data existing as a list of URLs, titles, and timestamps leaves it much neglected and difficult to semantically explore. SBERT embedding based approaches for creating an interpretable topic modeling space within which these documents can be semantically explored show promise as useful tools for information retrieval and knowledge discovery. However, like most unsupervised models, their performance is extremely sensitive to the quality of the input data. In the case of web page data, it is difficult to control for the quality of content, which leads to end topic models that include a significant amount of junk data. In this project, I explore PU learning and four different supervised learning models to build a classifier that automatically discards such undesirable web page documents from my topic modeling pipeline.

1 Introduction

Nowadays, much intellectual exploration happens through a web browser. However, the breadcrumbs that trail all this activity are largely unstructured; the tabular nature of this data existing as a list of URLs, titles, and timestamps leaves it much neglected and difficult to semantically explore. Topic modeling and document clustering are techniques used to manipulate and search collections of text for information retrieval and potential knowledge discovery. To address this, I have been building a tool that creates an interpretable document search space from a set of web page URLs. The use case that inspired this was a semantically explorable representation of my own browsing history— a mind map, in other words. Last quarter in CS 224N, I built a pipeline for constructing an interpretable document space and clustering the URLs into "topics" (discrete groupings based on subject) using Sentence-BERT (SBERT) embeddings [1] and HDBSCAN [2]. This method qualitatively proved to produce excellent topic groupings, especially given the diverse and unpredictable nature of the dataset [3]. [See Figure 1 for example].

While the representational pipeline for this shows promise, it is extremely sensitive to the quality of input data. To create the document embeddings for each URL, the web page data must first be downloaded and the meaningful text extracted from it. This in itself is challenging to fully automate since there is not a perfect way to discern meaningful user content from an HTML file. However, the Trifilatura library [4] does an excellent job doing this initial extraction of useful text from the web pages. While Trifilatura can extract reasonable user-facing content, there are still unfortunately many web pages that do not contribute meaningful information to the document clustering maps. This includes sites that require logins, pay-walled content, broken websites, a prompt to use flash player, invalid links, dynamic media feeds, and sites with no extensive text content. These websites consist of somewhere between $25\% - 50\%$ of my browsing history, and having them in the pipeline poses several challenges. Ideally, the identification and removal from the pipeline of these invalid web page documents would be automated. For the scope of this project, I aim to leverage machine learning classification methods to perform this discrimination. The classifier would accomplish the following:

- **Improve Data integrity:** Having these invalid URLs as data input to the unsupervised learning pipeline surely has an impact on the quality of the output. They pollute later dimensionality reduction of embeddings, and additionally find their way into otherwise useful clusters.
- **Reduce Extraneous Compute Costs:** There are many invalid web pages, and pushing each of these web pages through the later stages of the pipeline, such as clustering, increases the complexity and runtime of these steps.
- **Automate corpus pruning:** My current technique for discarding such invalid websites consists of a simple exclusion list. This exclusion list was updated only when I realized that certain websites needed to be excluded, oftentimes not until later in the pipeline. As new websites are added to the browsing history, validating them by hand would be necessary, which requires tedious work.

1.1 Related Work

Binary classification of document content is a well-studied task, particularly for the application of email spam filtering [4]. Methods employed for this type of task span from simple n-gram-style models and simple logistic regressors to more complex deep learning models.

2 Method and Experiments

The goal of this project is to build a binary classifier model that is effective at identifying web page documents that should be excluded from the topic modeling pipeline. Web page documents that meet this exclusion criteria include:

- **Broken Website:** Websites that no longer have content, content loading errors.
- **Login/Paywall blocked content:** Upon initial surveying, this is the largest group of invalid web pages. This includes things like Slack conversations, Canvas, Stanford Axess, Piazza, or anything else that is protected by a login and inaccessible to a web scraper.
- **Dynamic Content Feeds:** This includes front-page news URLs and other content feeds such as reddit.com. These websites have no static content and are not likely to be meaningful to the clustering process.

2.1 Data and Incomplete Labels

The dataset used for training this classifier is my own personal desktop browsing history data. While it contains 45k unique URLs, only 31k could be fetched by Trifilatura. These 31k web page documents are the inputs to the classifier.

Because this dataset is so large, it would be difficult to hand-label each example as to be included/excluded. Instead of hand-labeling the entire dataset, I employ Positive-Unlabeled learning, a type of binary classification where the training set consists of a set of positively-labeled examples and an additional unlabeled set that contains positive and negative examples in unknown proportions [5]. In this scenario, the "positive" examples are web pages from URL domains identified in my exclusion list. Applying this exclusion list, 9428 out of the 31k web page documents are flagged as invalid.

The exclusion list was hand-developed over the course of the previous project and includes 34 domain names. Some examples include domains such as `reddit.com, canvas.stanford.edu, slack.com, docs.google.com`.

2.1.1 Features

The inputs to the machine learning classification models are SBERT embeddings of the web page documents associated with each URL. The details of this embedding creation is specified in Appendix section A.1. This is a natural choice of featurization since many text-based learning models require an embedding of the text and these embeddings are already created as the backbone to the topic modeling pipeline that this classifier will serve. The embeddings for each web page are $\in \mathbb{R}^{768}$.

In addition to the SBERT embeddings, I added three new features after exploring the data by hand: number of lines in document, average number of words per line, and total number of words in document. These were selected due to the fact that many of the web pages I wish to exclude tend to be short, especially in the case of "invalid page" or "please sign in" pages.

2.2 Supervised Learning Models

- **Logistic Regression Classifier:** This is a simple linear classifier algorithm that optimizes the training loss $J(\theta) = \frac{1}{n}\sum_{i=1}^{n} -(y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))$, where h_θ is the model prediction output (logistic function). An L2 penalty is also applied for regularization.
- **Support Vector Machine (SVM):** A linear classification algorithm that optimizes the margin, or line of separation between the two classes. I use the Radial basis function kernel (RBF Kernel): $\exp(-\gamma||x-x'||^2)$ where $\gamma = \frac{1}{2\sigma^2}$, and x and x' are two samples.
- **Random Forest:** An ensemble method for classification that leverages several decision trees to produce decision rule-based predictions.
- **Neural Network:** This is a more advanced model with layers of weights and non-linear activations that learn complex features from the input data to make classifications. I use 3 hidden layers (sizes 200, 100, 20), Adam optimization, and ReLU activations. Deep learning methods are considered the most successful for tasks in spam document classification and are leveraged by most modern email services [4].

As discussed in the previous section, PU learning is being employed given its suitability to the dataset. Implementations for PU learning in this project were accomplished through the `pulearn` Python package, which provides wrappers for Scikit-learn classifier models.

3 Results and Analysis

3.1 Quantitative Evaluation: Performance on New Invalid Web Pages

For evaluation of the model, great caution had to be taken with the construction of training and test data. In order to evaluate how well this classifier could generalize, it was important to entirely hide a subset of known invalid web pages from the model's training. This was accomplished by selecting a random subset of the exclusion list URLs and dedicating them to only be seen at training time. All of the unlabeled URLs are then randomly split into train and test with probability $p = 0.15$. Had we simply randomized across all URLs, then the model would've seen many of the same web pages we are looking to generalize to during training (For example, almost all of duckduckgo's web pages only consist of "You are being redirected to the non-Javascript site". If a subset of these were included in training, then we would not truly be evaluating meaningful generalization during test time if other duckduckgo pages show up.).

Thus, when evaluating the different models, the overall test accuracy matters less overall than the model's test accuracy on correctly identifying items from the exclusion list. In Figure 1, this is shown in the "Exclusion only" column.

Model results: Accuracy by task and model				
Model	Train	Train (Exclusion only)	Test	Test (Exclusion only)
Logistic Regression	99.144	97.544	**98.271**	97.580
SVM	99.131	96.464	42.771	6.184
Random Forest	99.205	96.439	42.850	6.253
Neural Network	99.465	98.551	97.396	**98.553**

Figure 1: Results of the different models on the training and test sets. The "exclusion only" columns show the model's accuracy on correctly identifying the URLs labeled as invalid from the domain exclusion list.

3.2 Qualitative Evaluation

Since the primary motivation of this binary classifier is to improve the quality of downstream topic modeling tasks, I also consider a brief qualitative evaluation of the downstream topic modeling cluster quality. To do this, the neural network model classification predictions were used to discard about 9.5k web page documents from the embedding pipeline. I compared these end clustering results with that produced by the full dataset and saw that with these invalid web pages excluded, the clustering space was much more conducive to higher quality clusters. This is likely due to the fact the downstream stages of the pipeline first involve a dimensionality reduction, upon which the clustering takes place. Without the junk web pages, the dimensionality reduction step can represent more of the variance within the web pages that actually matter.

3.3 Discussion

Notably, the SVM and Random Forest models did not perform well on the exclusion-set-only URLs. When further inspecting this, a large contributing factor to this is that the domain duckduckgo was included in the test set (for all four models), and both models failed to classify any page from duckduckgo as invalid. On the other hand, the logistic regression and neural network models were able to learn the sufficient information to make this discrimination. Some surprising domains that the Neural Network and Logistic regression models did poorly on detecting were links from facebook.com and linkedin.com. Both of these websites only have documents with a variant of "please sign in to continue," but surprisingly the models were unprepared to make this distinction.

Additionally, the surprising aspects of the models' performances may be explained by the sparsity of the data. With only 34 hand-curated domain names to extrapolate off of, only about 21 got used during training, which may not be nearly enough to cover the full complexity of the website content I am looking to exclude.

On the other hand, the logistic regression and neural network models did manage to classify many unlabeled data points as invalid. Upon further inspection, these such websites were content that I was looking to exclude, but would've never thought to add to the exclusion list (since I simply had not noticed them before). Because of this, even though these models are not perfect, they still can be useful tools for further exploring the dataset and understanding aspects of it that could be useful in building future models.

4 Conclusion

4.1 Summary of Results

From this project, machine learning based classifiers for discarding invalid web page content from topic modeling pipelines show to be a promising method. Most critically, this project underlines the importance of understanding the data at hand. Approaching a large unlabeled dataset like this was a huge challenge, and there is still much to explore for improvement. Some of these ideas are laid out in the future research directions section.

4.2 Future Research Directions

Different approaches to labeling: In the presented method, only domain names were specified for the creation of exclusion labels. The upside to this is that a single blanket exclusion on a domain name can catch many variants. On the other hand, it is also the case that many websites under the same domain name may all give the same error message and thus not contribute anything new. Additionally, there are many instances where some URLs under the same domain will be valid, while others should be excluded. An alternative approach to generating training labels would be to label individual URLs, rather than blanket label from domain names.

More classification categories: Notably, login/paywall blocked content tends to be short and looks very different from how a dynamic news feed might render. Clumping both of these into one category may pose challenges. Additionally, adding multiple categories would allow for greater flexibility for the user to specify types of web content that is not meaningful to them in a mind-map (e.g. shopping).

Fine-tuned SBERT classification and other advanced neural network models: While this project demonstrated the promise of relatively simple neural networks for this classification task, there are much more sophisticated neural network classifiers out there. For example, SBERT could additionally be fine-tuned on input sentences from the documents to make classifications. However, this would require more training data. An observation that would favor these types of approaches is that the first few lines of a document include a lot of information about how likely it is to be invalid or not (e.g. error messages or sign in prompts are often very short). This would mean that a model could be constructed such that it only considers the first part of the document, from which it decides if it is invalid or not.

References

[1] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[2] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. 10(1), 2015.

[3] Natalie Cygan. Sentence-bert for interpretable topic modeling in web browsing data, 2021.

[4] Emmanuel Gbenga Dada, Joseph Stephen Bassi, Haruna Chiroma, Shafi'i Muhammad Abdulhamid, Adebayo Olusola Adetunmbi, and Opeyemi Emmanuel Ajibuwa. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 5(6):e01802, 2019.

[5] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220, 2008.

A Appendix

A.1 SBERT Embedding creation

Since SBERT has a limitation to the input text size (512 characters, enough for most sentences, short paragraphs), my procedure for constructing a document embedding involves tokenizing the entire document by sentences and then truncating any sentence that is too long (over 512 characters). Next, I use the SBERT model to compute a sentence embedding for each of these sentences. Finally, I average the embeddings for all sentences in the document to form a document embedding. This is repeated for each document/webpage in my dataset. [3].