Predicting and Identifying Hypertext in Wikipedia Articles

Neel Guha, Annie Hu, and Cindy Wang
{nguha, anniehu, ciwang}@stanford.edu

I. Introduction

Wikipedia now has millions of articles online and tens of millions of views each hour. In order to maintain a high standard of organization across this sheer volume of content, it is essential to ensure accurate and helpful intra-article linking. The Wikipedia Manual of Style states that links represent "major connections with the subject of another article that will help readers to understand the current article more fully."[1]

In this project, we optimize article linking towards this goal by applying machine learning to predict hypertext in Wikipedia articles (see Figure 1 for an example of hypertext). As inputs, our algorithm processes tokens (words and n-grams) from articles. We then use Naive Bayes, logistic regression, and a SVM to output a predicted classification: plain text (no link) or hypertext (link). In this paper we describe the process for our data collection, feature selection, and the relative performance of various algorithms. Note that we have limited the scope of this project to predicting solely hypertext and not hyperlinks. While there has been considerable work on predicting hyperlinks (especially using context resolution based techniques), we felt there had been significantly less work on predicting hypertext.

Our model enables large scale automated editing of Wikipedia linking in a low cost but effective way and can also be used to make link suggestions to individual contributors. Outside of Wikipedia, our system can be generalized to any broad database of documents or texts. For example, it could be applied to predict hypertext in news corpora or academic literature.

John Leroy Hennessy (born September 22, 1952) is an American computer scientist, academician, and businessman.

Fig. 1. In this sentence, the terms 'American' and 'academician' are hypertext.

[1]http://en.wikipedia.org/wiki/Wikipedia:Manual of Style

II. Related Work

There is considerable prior work involving using existing Wikipedia links for entity disambiguation [1]. Rather than general applications, the goal of our paper is to identify semantically important hypertext within the articles themselves.

Prior approaches to the problem of hypertext detection rely heavily on keyword identification algorithms. Generally, these approaches score potential keywords by evaluating 1), word occurrence statistics or 2), linking patterns on similar pages. The traditional statistical approach, which depends solely on word occurence counts, involves methods such as tf-idf, χ^2 independence test, and "keyphraseness" (proportion of occurrences that are hypertext) across the training set [2]. These methods have moderate success, with upper precision/recall/F_1 scores in the 0.5-0.6 range. The second approach involves semantic analysis to calculate metrics such as contextual commonness and relatedness. [3] and [4] outline specific formulas to evaluate these measures. [5] introduces a novel algorithm (LTRank) to identify similar pages, then identify candidate links from those similar pages that might be missing on the given page. The state-of-the-art algorithm for link detection has a precision and recall of approximately 0.75 [4].

We aim to develop a better model by incorporating not only statistical and contextual features, but considering the absolute importance of each word itself; i.e. proper nouns are more likely to be relevant links. [6] suggests a computational linguistics approach to keyword extraction involving part of speech (POS) tagging in additional to statistical features. We explain our feature selection in more detail below.

III. Data Set and Features

For clarity, we define "hypertext" as a word in an article that appears, in the article content, as a clickable link to another article. Our algorithm works as follows:

1) Model Training: For each article a_i in a training set of articles A, we we extract each unique token (word) t_i from a_i and represent a_i as the set of unique

tokens it contains. For each token we then construct a feature vector x_i based on the attributes of t_i relative to a_i (its parent article) and A (all the of articles). If in a_i, t_i is a word with a link to another Wikipedia article, we label t_i's output vector y_i as hypertext (y=1). Otherwise we label it as plaintext (y = 0). The set of all x_i, y_i's across all a_i in A forms our training set.

2) Model Testing: When testing this model on an article a_T, we follow a similar initial procedure. We extract each unique token from a_T and construct its corresponding feature vector. We then use the model we trained above to classify the set of feature vectors corresponding to the tokens in a_T. If a feature vector x_i is classified as hypertext, then its corresponding token is predicted to be linked on in the article a_T.

Because Wikipedia only links on the first occurrence of a word (as opposed to every occurrence of the word), note that we represent each article by the list of unique tokens it contains.

A. Features

To determine whether a word should be linked, we thus considered features pertaining to the word itself, as well as the word in the context of the article and data set as a whole. The four features included in our model are:

1: Whether or not a word is proper noun, 0 or 1. Proper nouns typically denote significant people, places, or things, which usually have their own Wikipedia pages. If these words appear in an article, they should be linked.

2: Length of the word, an integer value. Longer words are more likely to be less common, names, or technical terms. In any of these cases, the word should be linked.

3: The tf-idf score of the word (a float value between 0 and 1) relative to the article in which the word appears in. Tf-idf, or term frequency-inverse document frequency, is a numerical statistic that denotes how important a word is to a document. Specifically, for a term t in a document $d \in D$, $\text{tfidf}(t,d) = tf(t,d) \times idf(t,D)$ where $tf(t,d)$ is the normalized frequency of term t in document d and $idf(t,D)$ is the log of the proportion of the number of documents in D containing the term t.

4: The proportion of [the number of articles in which the word is linked] to [the number of articles in which the word is mentioned but not linked], a float value. This gave us a measure of how often a word was linked if it was mentioned.

B. Data set and data collection

The words that constitute links to other pages differ significantly from page to page and are highly dependent on the context of the page. Entities in an article that are more closely related to the topic of the article are more likely to constitute links then entities that are mentioned in passing. For example, in the Wikipedia article on Paul Ryan, the entity "marathon" is mentioned but does not link to its corresponding wikipedia page. However in the page for "Running", "marathon" is mentioned and links to its corresponding Wikipedia page. Because of this, attempting to predict hypertext across a set of articles covering diverse topics can be tricky. With such diverse data, it can be hard to determine meaningful conclusions on which terms should and shouldn't constitute links.

We hypothesize that accurately predicting hypertext must leverage an implicit understanding of the article's context and topical focus. We tested this by running our model on two different data sets.

1: Intuitively, articles from the same Wikipedia category should have similar distributions of hypertext. Limiting the articles in the training and testing set to one category thus allows the model to implicitly account for the broader topic/context of the articles. In this project, we selected random samples of 100 training articles and 30 test articles from the Wikipedia "Forms of Government" category. We called this our government dataset.

2: In order to test our model in a case where the articles in the dataset do not originate from the same context, we ran our model on a 600 training articles and 200 test articles uniformly drawn from a mixture of 4 Wikipedia categories ("Judaism","Ancient Greece","Environmental science","Technology"). We called this our multicategory dataset.

We scraped and parsed each of these articles using BeautifulSoup. For each article we extracted a list of unigrams/bigrams as well as a list of terms that are linked on.

We evaluated a variety of possible features to model. We treated each unique word in a given article as a separate training example.

Creating the feature vectors for the training set required significant pre-processing of our data. We utilized the NLTK and scikit libraries to perform part of speech classification and tf-idf scoring, respectively. We applied NLTK part of speech tagging to the tokenized plaintext of each article, and filtered the

words tagged as proper nouns. We then used scikit tf-idf term weighting to vectorize and score each word in the context of the article in which it appeared. We wrote custom scripts to determine the other features.

IV. Methods

In our algorithm, we experimented with the following models:

A. Dummy Model

In order to establish a baseline for our algorithm's performance we used a dummy stratified classifier from the sklearn library [6]. This classifier establishes a baseline performance by randomly predicting a class for each training example. Because our data set contains a fairly significant class imbalance (the number of terms that aren't links far exceeds the number that are), the probability of picking each class was weighted by the class distribution in the training set.

B. Gaussian Naive Bayes

Naive Bayes models are useful for quick and initial evaluations of a dataset to explore the feasibility of building some classifier. Though they make the "naive" assumption of conditional independence between features, they've proved to be surprisingly reliable. For a y_i with with an associated set of features $x_0, ...x_n$:

$$P(y_i|x_1,...,x_n) = \frac{p(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1,...,x_n)} \quad (1)$$

We first implemented a Gaussian Naive Bayes model where the feature likelihood is assumed to follow a Gaussian distribution.

C. Logistic Regression

We used a logistic regression model with l2 regularization [7]. In this type of classifier the hypothesis function takes the form of a logistic curve:

$$h_\theta(x) = \frac{1}{1+\exp(-\theta x)} \quad (2)$$

where $h_\theta(x)$ can be viewed as the probability $y = 1$ given an x. We used an l2 penalty with logistic regression.

D. Support Vector Machine

Support Vector Machines attempt to classify data by constructing a multidimensional hyperplane to separate data points of differing classes (in effect identifying a decision boundary). The ideal decision boundary is given by a hyperplane which is at the maximum distance from the closest points of each class (referred to as support vectors). In the case where the data is not linearly separable, SVMs utilize Kernel functions to project the data into a higher dimension where a linear separator can be found. Specifically, SVMs operate by solving the optimization problem:

$$\max_\alpha W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j\langle x^{(i)}, x^{(j)}\rangle \quad (3)$$

$$\text{s.t.} \quad \alpha_i \geq 0, i = 1,\ldots,m \quad (4)$$

$$\sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \quad (5)$$

V. Results and Discussion

A. Data

The government category dataset yielded 148,303 training feature vectors and 43,310 test feature vectors. For both training and testing, around 36-37% of the vectors corresponded to a linked word. The multicategory data yielded 625,826 training feature vectors and 214,912 test feature vectors. For both training and testing, around 31-32% of vectors corresponded to a linked word.

The Vovinam and Bình Định martial arts are widespread in Vietnam, while soccer is the country's most popular team sport. Its national team won the ASEAN Football Championship in 2008. Other Western sports, such as badminton, tennis, volleyball, ping-pong and chess, are also widely popular.

The Vovinam and Bình Định martial arts are widespread in Vietnam, while soccer is the country's most popular team sport. Its national team won the ASEAN Football Championship in 2008. Other Western sports, such as badminton, tennis, volleyball, ping-pong and chess, are also widely popular.

Fig. 2. Hypertext in original article (top) vs. predicted hypertext from model (bottom).

Our confusion matrix after running logistic regression on the government category dataset was:

	Predict NO	Predict YES
True NO	0.9287	0.07135
True YES	0.4433	0.5567

Although the model very accurately predicted true non-linked words, over 44% of predictions were false negatives, or words predicted as non-links that should actually have been links. A large majority of these false negatives were common words, such as "music", that were linked in the article as part of a multi-word phrase, such as "Vietnamese music". We hypothesized that updating our model to

consider bigrams, and eventually general n-grams, should eliminate much of these false negatives.

Our confusion matrix after running logistic regression on the multicategory dataset was:

	Predict NO	Predict YES
True NO	0.93251087	0.06748913
True YES	0.53432279	0.46567721

Even though we expanded the size of our dataset and diversified the categories we considered, the proportions of true positives and true negatives were very similar between the two datasets. Again, we hypothesized that considering single word tokens instead of n-grams caused much of this similarity. The imbalance in our linked and non-linked classes may also have contributed to the high false negative rate.

B. Metrics

Given the significant class imbalance in our data set, we evaluated our algorithms' performance by measuring their accuracy, precision, recall, $F1$, and AUC score. Precision is calculated as the number of correctly identified hypertext tokens divided by the total number of hypertext tokens proposed by the system; recall is defined as the number of correctly identified hypertext tokens divided by the total number of hypertext tokens in the original document; and F_1 score is the harmonic mean of the precision and recall. AUC is defined as the area under the ROC curve, which is created by plotting the recall against the false positive rate (the number of incorrectly identified hypertext tokens divided by the total number of non-hypertext tokens) at various threshold settings.

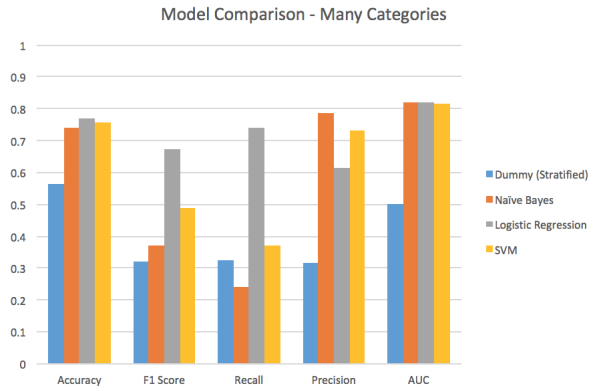

Fig. 3. A comparison of the performance of various models for the mixed categories data set

C. Comparative model performance

The results of the various models along each of the metrics mentioned above for the government data set and the multi-category data set can be seen in Figure 4 and Figure 3 respectively. As expected, the dummy model performed very poorly and was surpassed by every other model.

Within a single category, logistic regression and SVM consistently outperformed Naive Bayes across all metrics. Naive Bayes likely performs poorly because of the independence assumptions it makes on the features. Whether a word is a proper noun is likely strongly correlated with the proportion of times it is linked when mentioned, particularly within a set of related articles. For instance, the proper noun "Vietnam" was linked in most of the articles in which it appears. Given our feature set, logistic regression and SVM seem to have reached an upper threshold in terms of performance. Using articles across several categories, accuracy and AUC values for all models were roughly the same. However, SVM performed considerably worse, while logistic regression performed considerably better across most metrics. SVM most likely suffered in the multi-category case since the classes became less separable. Consider important technical words that are mentioned few times in one article but many times in articles from other categories, resulting in high word length and low tf-idf score. Now consider short names of important historical figures, which have low word length but high tf-idf score. Since both types of terms are good candidates for hypertext, this illustrates how the classes are difficult to separate with a (n-1)-dimensional hyperplane. Since these instances are more likely to occur in the multi-category case, SVM has a much lower F1 score and recall. Thus, logistic regression clearly performs best when the data is not restricted to semantically related articles.

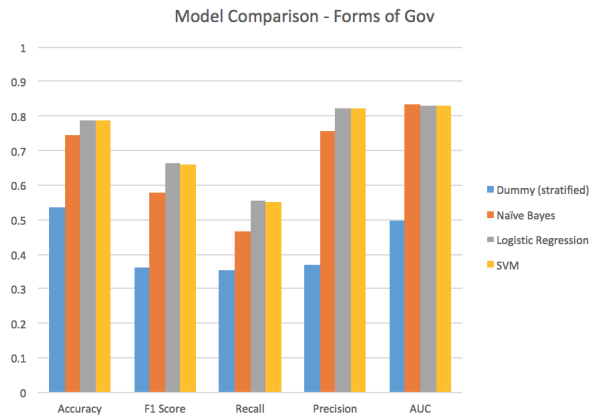

Fig. 4. A comparison of the performance of various models for the government data set

D. Feature Analysis

To analyze which of our features were most relevant to our predictions, we performed a simple Leave-One-Out feature analysis on each of our main models. For each feature, we retrained and retested the model with that feature excluded, and compared the resulting metrics across features.

For the government dataset, word length was actually our most relevant feature, with tf-idf values and hypertext

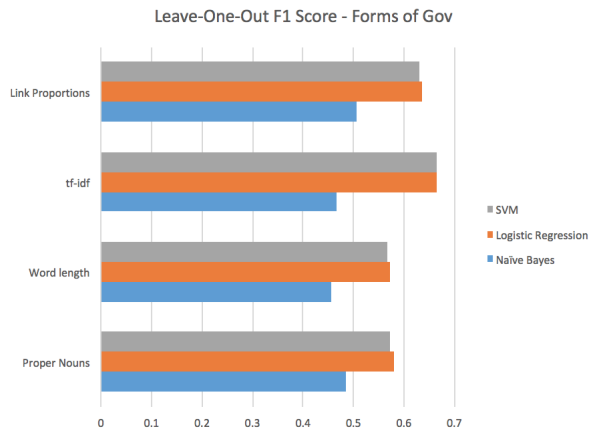

Fig. 5. LOO Feature Metrics for Government Dataset

proportions much less relevant. This seemed to suggest that hypertext depended only loosely on article context, directly contradicting our earlier hypothesis. Since we limited input vectors for this dataset to one category of articles, the articles may have been *too* similar, limiting the usefulness of these two features in differentiating context. The similarity of article context combined with our relatively small sample size may have also caused overfitting to our training set, and an inflated accuracy of prediction. To further explore these two observations, we then ran Leave-One-Out feature analysis on our multicategory dataset.

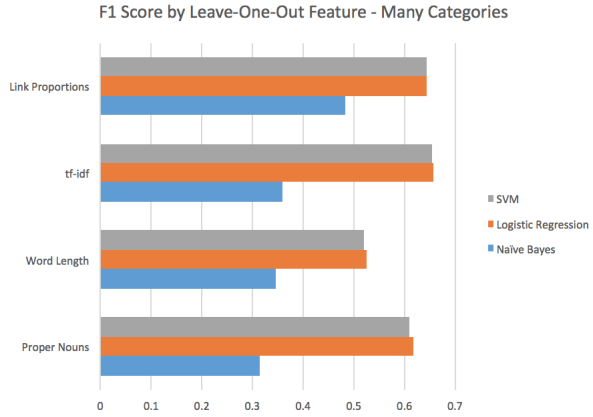

Fig. 6. LOO Feature Metrics for Multicategory Dataset

Surprisingly, our results did not change much even after considering multiple categories – in fact, word length became even more relevant to our results. Though prior approaches to predicting hypertext tended to focus on statistical measures of word importance, such as tf-idf, our results when mixing these measures with linguistic features such as word length seem to suggest that linguistic features correlate much more strongly to predictions.

E. Experimenting with Bigrams

For the government data set we also experimented with running our algorithm on bigrams. For each article - in addition to creating feature vectors for single word tokens - we create feature vectors for bigrams (pairs of adjacent words). Bigrams are frequently used in many natural language applications and have been shown to be effective in a wide range of problems. Though all of our models outperformed the dummy model when run on bigrams, there was significantly less variance between the models (see 7). In fact, all of the models performed significantly poorer on the bigram dataset then on the single word dataset.

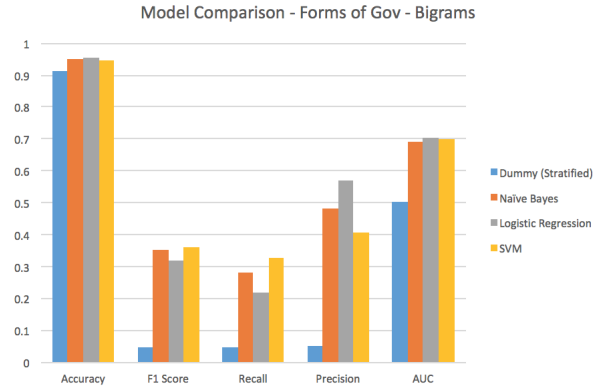

Fig. 7. A comparison of the performance of various models for the government data set using bigrams

However, our resulting confusion matrix from running logistic regression did significantly reduce our number of false negatives, as predicted:

	Predict NO	Predict YES
True NO	0.8579	0.1421
True YES	0.0804	0.9196

VI. Future Work

In this project we described an algorithm for predicting and identifying hypertext in Wikipedia articles. The logistic regression and SVM algorithms perform best in the case of a single-category data set. However, across multiple categories, logistic regression definitively outperforms all other algorithms, reaching .77 accuracy and .81 AUC.

Possible next steps could include expanding the data set and identifying more features to help decrease false-negatives. We had computational difficulties (limited RAM) in implementing n-grams in this project, and were only able to achieve mediocre results with bigrams on a small data set. With greater resources we would hopefully be able to complete a more thorough implementation of n-grams. This too could improve our results.

VII. References

[1] Ratinov, Lev, et al. "Local and global algorithms for disambiguation to wikipedia." Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

[2] Mihalcea, Rada, and Andras Csomai. "Wikify!: linking documents to encyclopedic knowledge." Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. ACM, 2007.

[3] Gabrilovich, Evgeniy, and Shaul Markovitch. "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis." IJCAI. Vol. 7. 2007.

[4] Milne, David, and Ian H. Witten. "Learning to link with wikipedia." Proceedings of the 17th ACM conference on Information and knowledge management. ACM, 2008.

[5] Adafre, Sisay Fissaha, and Maarten de Rijke. "Discovering missing links in Wikipedia." Proceedings of the 3rd international workshop on Link discovery. ACM, 2005.

[6] Hulth, Anette. "Improved automatic keyword extraction given more linguistic knowledge." Proceedings of the 2003 conference on Empirical methods in natural language processing. Association for Computational Linguistics, 2003.

VIII. Tools

[1] NTLK Library. Bird, Steven, Edward Loper and Ewan Klein. Natural Language Processing with Python. O'Reilly Media Inc. 2009.

[2] scikit-learn. Pedregosa et al. Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830, 2011.