LLAMA: Automatic Hypertext Generation Utilizing Language Models

Dong Zhou
University of Nottingham
School of CS and IT
dxz@cs.nott.ac.uk

James Goulding
University of Nottingham
School of CS and IT
jog@cs.nott.ac.uk

Mark Truran
University of Teesside
School of Computing
m.a.truran@tees.ac.uk

Tim Brailsford
University of Nottingham
School of CS and IT
tim.brailsford@cs.nott.ac.uk

ABSTRACT

Manual hypertext construction is labour intensive and prone to error. Robust systems capable of automatic hypertext generation (AHG) could be of direct benefit to those individuals responsible for hypertext authoring. In this paper we propose a novel technique for the autonomous creation of hypertext which is dependent upon *language models*. This work is strongly influenced by those algorithms which process the hyperlinked structure of a corpus in an attempt to find authoritative sources. The algorithm was evaluated by experimental comparison with human hypertext authors, and we found that both approaches produced broadly similar results.

Categories and Subject Descriptors

H.5.4 [**Information Interfaces and Presentation**]: Hypertext / Hypermedia - *Architectures.*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing - *Linguistic processing.*

General Terms

Algorithms, Experimentation, Languages, Measurement, Theory

Keywords

Automatic Hypertext Generation, Bipartite Graph, Clustering, HITS, Language Models

1. INTRODUCTION

The effort required to manually hyperlink a collection of text files is prodigious. Any software tool capable of *reliably* authoring hyperlinks without human intervention would be highly desirable given the arduous and time-consuming nature of hypertext creation [4]. However, attempts to develop such a tool have proven it to be an tremendously challenging task [1, 2]. Consequently, research and development of Automatic Hypertext Generation (AHG) systems has diminished, and promising work has become mired in the difficulties of emulating a very subjective process - *How does a human approach linking? Do humans link the same way? How do we assess the links produced by an authoring tool? How do we prevent over-authoring?*

In contrast, the related discipline of Information Retrieval (IR) has moved forwards apace, gaining maturity and depth. Approaches such as the application of language models to text analysis have proffered significant results [7], while methods for retrieving and ranking documents, such as HITS [6] and PageRank [3], have proven to have extremely effective real-world applications.

Therefore, we now present a novel automatic hypertext generation algorithm which incorporates leading edge information retrieval techniques - statistical language models, graph based algorithms, inter-feature relationship modelling and cluster based analysis. Experimental results are particularly encouraging, with initial findings indicating that our system and human authors have very close overall performance in generating links.

2. OVERVIEW OF THE SYSTEM

In generating hyperlinks for a collection of raw text documents, our algorithm invokes several systematic steps. These steps are described *in detail* in section 3. However, a *simplified* description of the process is as follows:

- The first step is to identify important *text features* in the document collection that can serve as anchor and target points for the links we generate. We identify these *dominant* (target) and *subordinate* (anchor) features through the application of language models.
- The second step involves filtering out the weaker subordinate features to obtain a set of the most credible anchors. Through graph based analysis, our algorithm then matches the remaining anchors with the most appropriate endpoints, with the dominance of possible targets determining suitability. It is from these matches that the final links are produced.

HT'07, September 10–12, 2007, Manchester, United Kingdom.

In implementing this process, we are able to show that an authoring tool based on language models and graph based analysis is particularly suitable for automatically generating hypertexts from *raw texts*

3. DESCRIPTION OF LLAMA

The *Linking Language Models Algorithm* (LLAMA) is described in detail below. Throughout the rest of this paper we assume the following notation:

c : *a concept or topic*

t : *a single term*

d : *a document*

f : *a feature (a concept, single or multiple terms, a sentence, a paragraph in a document or even the document itself)*

C : *a cluster of documents*

LLAMA is reliant upon of the use of statistical language models. Henceforth, we denote the language model score for a feature x that is induced from feature y as: $LM_x y$.

In LLAMA, links are generated by examining the *dominance* and *subordinance* of features. These concepts bear a strong similarity to the determination of *hubs* and *authorities*, as proposed by Kleinberg [6]. In much the same way as hubs and authorities, dominant and subordinate features have a *mutually reinforcing relationship*. Thus the dominance and subordinance of features in a text document are defined recursively as follows:

1. A *strong dominant feature* is one that is *pointed at* by a high number of strong subordinate features.
2. A *strong subordinate feature* is one that *points to* a high number of strong dominant features.

In the LLAMA approach a feature can be any component of a document - a single term, a sentence, a paragraph, or even the whole document itself. However, throughout this paper, we accord to the general form of hypertext currently popularized by the Web - *dominant* features (those which we desire to link to) will be constrained to whole documents, and *subordinate* features (those we desire to link from) will be constrained to terms within documents. Because links constructed by LLAMA always flow from a *subordinate* feature to a *dominant* feature, the links created during our experiments will always be *from* terms and *to* documents .

It will also be valuable to define a metric to measure just how dominant or subordinate a feature actually is. For this purpose, we will use the score $Dom(f)$ to indicate the level of dominance attributed to a given feature f, and the score $Sub(f)$ to denote the level of subordinance of feature f.

3.1 The Link Generation Algorithm

1. Consider a collection of documents, C, or optionally create clusters of documents $\{C_1, C_2, , C_n\}$ offline.

2. Generate *all* possible feature links inside the collection C (or individual clusters, C_i) to form a directed weighted graph: $G =< F, W >$, where F is the set of vertices representing all features in the collection, and W is a complete set of *weighting functions*. Hence, every possible pairing of features has a non-negative weight attribute, w, which indicates the probable strength of any link potential between the two features. The set of weights as whole can be described as:

$$W : F \times F \longrightarrow \{w \in \Re : w \geq 0\}$$

An individual weighting between features f_1 and f_2 is given by the function:

$$w(f_1 \longrightarrow f_2)$$

(n.b. In LLAMA each of these weights are calculated using language models, but within this remit there are still two possible approaches, as discussed in next section)

3. For each feature, f, compute the *Dominance Score* and *Subordinance Score* of f in order to determine: $Dom(f)$ and $Sub(f)$ for every single feature in the collection (this process is discussed in more detail in next section).

4. Select a set of the strongest subordinate features, $Strongest_{Sub}$, from the collection C. This set can be formed in one of two ways: by retaining a fixed number of the features, β, with the highest $Sub(f)$ scores (as we have done in the experiments discussed in this paper) or by retaining only those features with a subordinance above a certain threshold, τ.

5. For each of the Subordinate Features that were retained, $f \in Strongest_{Sub}$, determine a measure of linkage fitness, $Fit(f, f')$, between it and all possible target features, f' (which in our experiments are constrained to be documents). This fitness can be measured using one of the following two metrics:

$$Fit(f, f^{'}) = LM_{f'} f \tag{1}$$

$$Fit(f, f^{'}) = LM_{f'} f \times Dom(f^{'}) \tag{2}$$

The first, which to refer to as the $Fit1$ function, uses the language model score between f and f' directly. The second, which we refer to as the $Fit2$ function, additionally incorporates the dominance of any target being considered.

6. The preferred link endpoint for each feature, $f \in Strongest_{Sub}$, is then determined by selecting the dominant feature, f', which produces the best fitness measure:

$$\langle f, f^{'} \rangle : Max(Fit(f, f^{'}))$$

It is worth noting that this will generate a single end link for each retained subordinate feature. However it would be possible to replace Max with a different determining function that returns a set of anchor, target couples. This would allow a feature, f, to be paired with more than one target feature, f', providing a mechanism for automatic multi-headed link generation.

7. Collate and output results.

3.2 Dominance, Subordinance, and Language Models

We now consider the nature of *Dominance, Subordinance* of a feature and *Language Models* in greater depth. In step 3 of the LLAMA algorithm we compute a score for a feature's dominance by projecting it over the rest of the collection under consideration. LLAMA defines the dominance of an individual feature, f, over the whole collection of features, F, to be:

$$Dom(f) = \sum_{f' \in F} w(f^{'} \longrightarrow f) \times Sub(f^{'}) \tag{3}$$

This formula provides a natural measure of how "strong" f is as a target for a link, taking into account how powerful the terms associated with it are themselves. Therefore a strong dominating feature is likely to be pointed to by high-quality subordinate nodes (which will, by definition, tend to point to "strong" dominating features themselves). In an exact mirror of this calculation LLAMA determines the strength of a subordinate feature as follows:

$$Sub(f) = \sum_{f \in F} w(f \longrightarrow f^{'}) \times Dom(f^{'}) \quad (4)$$

Clearly, equations 3 and 4 are mutually recursive. However, as with the iterative HITS algorithm[6], it can be proven that these equations will converge to score functions $Sub*$ and $Dom*$ (which are non-identically-zero, non-negative). When the LLAMA algorithm has converged, those features with the top Subordinance and Dominance Scores, or those above the preset threshold levels, will be natural choices for providing link anchors and targets.

We now turn our attention to how we use language models to calculate a score for the association between two features. Language models are based on the assignment of probabilities to text sequences in order to measure the strength of association between different components of the corpus. Given a context d in which term t occurs, a raw probability is appointed to a single term t using Maximum-Likelihood Estimation (MLE):

$$MLE_d t = \frac{g(t,d)}{\sum_{t'} g(t',d)} \quad (5)$$

Here the denominator $g(t,d)$ denotes the number of times the term t occurs in context d, and the numerator denotes the number of times all terms appear in d.

The MLE used by language models is typically extended to distributions over term sequences, such as sentences, by assuming that the terms contained therein are independent. For an n-term text sequence we thus have:

$$MLE_d t_1, t_2, \cdots, t_n = \prod_{j=1}^{n} MLE_d t_j \quad (6)$$

The next step is to turn these raw probability values into an estimate of the strength of association between two different text sequences. The probability estimator that our approach uses is based on the *Kullback Leibler Divergence*[5], and defined as:

$$KL(d \parallel t_1, t_2, \cdots, t_n) = \sum_{t} MLE_d t \log \frac{MLE_d t}{MLE_{t_1,t_2,\cdots,t_n}} \quad (7)$$

The reason we exploit KL divergence is motivated by our desire to capture the asymmetric characteristics of the underlying graph-based framework. Weighting within LLAMA requires that association scores be uniformly positive, so we finally adjust the weight between two features to be:

$$LM_{f'} f = \exp(-KL(f \parallel f^{'})) \quad (8)$$

LLAMA only allows a feature to receive weights from other features when the association between them is high, otherwise the weighting is set to zero. The weighting function, $w(f_1 \longrightarrow f_2)$, may itself consist of two approaches, which we call *StrengthWeighting (SW)* and *FixedWeighting (FW)* respectively:

StrengthWeighting : If a target feature, f', is inside the nearest neighbourhood set of a given feature, f, we set the weights as:

$$w(f \longrightarrow f^{'}) = \begin{cases} LM_{f'} f & f \in nearest(f) \\ 0 & f \notin nearest(f) \end{cases}$$

FixedWeighting : If a target feature, f', is inside the nearest neighbourhood set of a given feature, f, we set the weights as:

$$w(f \longrightarrow f^{'}) = \begin{cases} 1 & f \in nearest(f) \\ 0 & f \notin nearest(f) \end{cases}$$

The nearest neighbourhood set $nearest(f)$ of a given feature is acquired by arbitrarily choosing parameter γ in order to produce the set of target features that yield the highest $LM_{f'} f$ where $|nearest(f')| = \gamma$.

In the experiments discussed within this paper prior clustering is applied to the collection, either via manual clustering or via hard clustering, using the bi-secting k-means algorithm[1].

4. EXPERIMENTAL DESIGN

The test collection for this study was manually constructed using the Wikipedia online encyclopedia[2]. Documents were taken from five arbitrarily selected subject categories: Arts, Sports, Economics, IT and Mathematics. Document length was capped at 500 words and each document discussed one single topic. A document set of 20 articles was extracted from each subject category.

Following article selection, mark-up tags and multimedia objects were extracted from the documents leaving a link-free text corpus. The next step involved applying a stop list of 571 words[3] to the document collection, stemming all document terms using Porter's algorithm[8] and indexing each document in the collection using the Lemur Toolkit[4].

During the experiment, the parameters β and γ were set to 8 and 15 respectively. These figures were selected after studying the characteristics of the collection. In total, we performed six authoring runs using six permutations of our main algorithm:

FW : LLAMA with *FixedWeighting* and *Fit1* function

FW* : LLAMA with *FixedWeighting* and *Fit2*function

SW : LLAMA with *StrengthWeighting* and *Fit1* function

SW* : LLAMA with *StrengthWeighting* and *Fit2* function

MC-FW : *FW* with prior manual clustering

HC-FW : *FW* with prior hard clustering

4.1 Evaluation of links

The first step in our assessment methodology involved constructing a *relevance judgment pool*. Every document in the test collection has a specific, one-term topic descriptor (i.e. the original article's title, removed prior to any

[1]taken from the Lemur IR project, available at http://www.lemurproject.org
[2]http://www.wikipedia.org
[3]ftp://ftp.cs.cornell.edu/pub/smart/
[4]http://www.lemurproject.org

processing by LLAMA). We added links to the relevance judgment pool that connected every occurrence of a term in the document collection to the article of the same name. This exhaustive process generated 774 ' likely correct' links. We belive that this is a reasonable method for filling the judgment pool, especially given the syntactic context of test data (i.e. internal Wiki links always have the name of an article as their anchor text).

We then evaluated the original Wikipedia article and the 6 LLAMA-authored versions of the same material using this relevance judgement pool. If a hyperlink existed in the Wikipedia original or the LLAMA-authored versions linking term x in document a to document b, and that link was duplicated in the relevance judgment pool, it was deemed 'correct' for the purposes of calculating precision/recall. We adapted these measures in the following manner:

$$precision = \frac{number \cdot of \cdot correct \cdot links \cdot generated}{total \cdot number \cdot of \cdot links \cdot generated}$$

$$recall = \frac{number \cdot of \cdot correct \cdot links \cdot generated}{number \cdot of \cdot correct \cdot links \cdot in \cdot the \cdot collection}$$

In our analysis we have also adopted the F-measure, which is defined as:

$$F = \frac{2PR}{P+R}$$

where P denotes precision, and R is recall.

4.2 Evaluation Results

Our experimental results are given in Table 1. Recall and precision scores indicate that links authored by human authors tend to have high precision but correspondingly low recall. LLAMA achieved a recall scoring equivalent to the Wikipedia authors, but was outperformed in terms of precision. However, from the F-measure, a combination of recall and precision, we can see that LLAMA and human authors have very close *overall performance* in generating links.

The interpolated precision-recall curves of all six permutations on our main algorithm are shown in Figure 1. Although the algorithms using *FixedWeighting* and *StrengthWeighting* performed identically, comparisons between the use of *Fit1* and *Fit2* functions in determining dominance and subordinance was not as clear cut. The clustering-based authoring runs performed poorly compared to the non-clustering variants.

5. CONCLUSION

In this paper we have introduced a novel and effective algorithm for the unsupervised creation of hypertext links within a raw text corpus. Experimental results suggest that our system and human authors have very close overall performance in generating links.

Future work could focus upon extending the algorithm above to consider phrases as link anchors. This could be accomplished quite simply by converting to a bi-gram or higher order language models. A second possibility is the extensibility of the algorithm to hypermedia objects other than text. Used in conjunction with voice recognition, it is possible in the future that this approach could enable hypertext functionality to be derived automatically from broadcast media in real time.

Table 1: Precision, Recall and F-Measure

Author	Precision	Recall	F-Measure
FW	0.34625	0.357881	1.073643
FW*	0.34625	0.357881	1.073643
SW	0.33625	0.347545	1.042635
SW*	0.33625	0.347545	1.042635
MC-FW	0.28375	0.293282	0.879846
HC-FW	0.20625	0.213178	0.639534
Wikipedia	0.890034	0.334625	1.003875

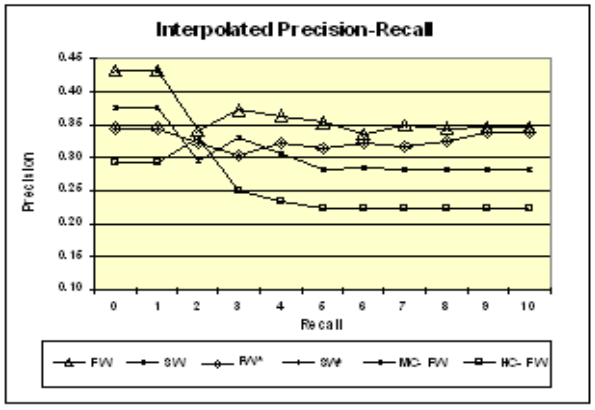

Figure 1: A comparison of six variations on the LLAMA algorithm

6. ACKNOWLEDGMENTS

We would like to extend our thanks to Dr. W. Lowe and Prof. C. Van der Eijk for the many invaluable discussions and advice offered throughout this study.

7. REFERENCES

[1] M. Agosti, F. Crestani, and M. Melucci. On the use of information retrieval techniques for the automatic construction of hypertext. *Inf. Process. Manage.*, 33(2):133–144, 1997.

[2] J. Allan. Automatic hypertext link typing. In *Proceedings of the the seventh ACM conference on Hypertext*, pages 42–52, Bethesda, Maryland, United States, 1996. ACM Press.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117, 1998.

[4] V. Bush. As we may think. *The Atlantic Monthly*, 176(1):101–108, 1945.

[5] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991.

[6] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[7] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, Melbourne, Australia, 1998. ACM Press.

[8] M. F. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980.