# LlamaFur: Learning Latent Category Matrix to Find Unexpected Relations in Wikipedia

Paolo Boldi*
Dipartimento di Informatica
Università degli Studi di Milano
Italy
paolo.boldi@unimi.it

Corrado Monti
Dipartimento di Informatica
Università degli Studi di Milano
Italy
corrado.monti@unimi.it

## ABSTRACT

Besides finding trends and unveiling typical patterns, modern information retrieval is increasingly more interested in the discovery of surprising information in textual datasets. In this work we focus on finding *unexpected links* in hyperlinked document corpora when documents are assigned to categories; our approach is based on the determination of a latent category matrix that explains common links; the matrix is built using a perceptron-like technique. We show that our method provides better accuracy than most existing text-based techniques, with higher efficiency and relying on a much smaller amount of information. It also provides higher precision than standard link prediction, especially at low recall levels; the two methods are in fact shown to be orthogonal and can therefore be fruifully combined.

## Categories and Subject Descriptors

H.2.8 [**Database Management**]: Data Mining; H.3.3 [**Information Storage and Retrieval**]: Information Retrieval; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods

## General Terms

Networks; Categorization; Outliers

## 1. INTRODUCTION

In general, data mining (text mining, if the data involved take the form of textual documents) aims at extracting potentially useful information from some (typically, unstructured, or poorly structured) dataset. The basic and foremost aim of data mining is discovering frequent patterns, and this problem attracted and still attracts a large part of the research efforts in this field. Nonetheless a quite important and somehow dual problem is that of finding unexpected (unusual, new, unforeseen...) information; it is surprising to note that this line of investigation did not receive the same amount of attention.

Albeit there is some research on the determination of surprising information in textual corpora (most often based on the determination of outliers in the distribution of terms or $n$-grams) there is essentially no work dealing with *unexpected links*. Even if some of the previous proposals exploiting text features can be adapted to this case, a simpler (and, as we here show, more effective) way to approach this problem is by using link prediction algorithms [15], stipulating that a link that is difficult to predict is unexpected.

In this paper, we prove that the availability of some form of categorization of documents can significantly improve the techniques described, leading to algorithms that are extremely efficient, use much less information than text-based methods, and offer better precision/recall trade-offs. Compared to link prediction, our technique also provides higher precision at low recall levels; moreover, the two methods have orthogonal outputs, and therefore their combination improves over both.

Our idea is that if the documents within a linked corpora are tagged with categorical information, one can learn which category/category pairs are more likely to appear, and as a consequence determine which links are unusual (in the sense that they are not "typical"). For example, documents of the category "Actor" often contain links to the category "Movie" (simply because almost all actor pages contain links to the movies they acted in). The fact that George Clooney used to own Max, a 300-pound pig, for 18 years presents itself as a link from an "Actor" page to a page belonging to the category "Pigs"/"Coprophagous animals", which is atypical in the sense above.

Our basic algorithm – henceforth called *LlamaFur*, "Learning LAtent MAtrix to Find Unexpected Relations" – tries to learn a category/category matrix describing the latent relations between categories: to this aim we apply a Passive-Aggressive learning method. Then we reconstruct which part of the graph is more explainable according to the matrix, and which links cannot be justified by the categories alone. Not only LlamaFur is also more efficient than both link prediction and the previous techniques based on the analysis of the textual content of the page, but it also improves the accuracy of link prediction algorithms in identifying unexpected links, if the two are combined.

*Partially supported by the EU-FET grant NADINE (GA 288956).

It is worth noting that the discovery of unexpected links offers a chance to find unknown information: given a certain document, we can highlight text snippets containing unexpected links. Meaningful text is often characterized, in web documents, by the presence of links that enrich its semantic; this is especially true in the case of Wikipedia, often used as a knowledge base for ontologies. Its link structure has proven to be a powerful resource for many tasks [20, 19]. For this reason, finding unexpected links seems a valuable way to detect meaningful text with information unknown to the reader.

Nevertheless, our results could be in principle applied to a plethora of different kinds of objects. The only assumption on the input is a (directed) graph, and a meaningful categorization of its nodes; categories can be overlapping as well, so in fact they may just be some observed features of each object. These assumptions are quite general, and could be applied to many use cases, from the detection of unexpected collaborations between grouped individuals to finding surprising travel habits from geotagged data.

The paper is organized as follows. In Section 2 we will review other works dealing with mining of unexpected information. Our technique will be presented through Section 3, where we explain how to estimate a latent category matrix with online learning; Section 4, where we describe a simpler, naive way to compute it; and Section 5, where we show how to use such a matrix to measure the unexpectedness of a link. In Section 6 we exhibit experimental evidence for the efficacy of our methods, by comparing them with different approaches derived from literature. Finally, in Section 7 we will sum up our work and suggest possible directions for future research.

## 2. RELATED WORK

One of the first papers trying to consider the problem in the context of text mining was [14]. In that work, two supposedly similar web sites are compared (ideally, two web sites of two competitors). The authors first try to find a match between the pages of the two web sites, and then propose a measure of unexpectedness of a term when comparing two otherwise similar pages. All measures are based on term (or document) frequencies; unexpected links are also dealt with but in a quite simplistic manner (a link in one of the two web sites is considered "unexpected" if it is not contained in the other).

This unexpectedness measure is taken up in [11], where the aim is that of finding documents that are similar to a given set of samples and ordering the results based on their unexpectedness, using also the document structure to enhance the measures defined in [14]. Finding outliers in web collections is also considered in [3], where again dissimilarity scores are computed based on word and $n$-gram frequency.

Some authors approach the strictly related problem of determining lacking content (called *content hole* in [18]) rather than unexpected information, using Wikipedia as knowledge base. A similar task is undertaken by [9], this time assuming the dual approach of finding content holes in Wikipedia using the web as a source of information.

More recently, [21] considers the problem of finding unexpected related terms using Wikipedia as source, and taking into account at the same time the relation between terms and their centrality.

An alternative way to approach the problem of finding unexpected links is by using *link prediction* [15]: the expectedness of a link $e$ in a network $G$ is the likelihood of the creation of $e$ in $G - \{e\}$. In fact, we will later show that state-of-the-art link prediction algorithms like [1] are very good at evaluating the (un)expectedness of links. Nonetheless, it turns out that the signal obtained from the latent categoy matrix is even better and partly orthogonal to the one that comes from the graph alone, and combining the two techniques greatly improve the accuracy of both.

## 3. LEARNING THE CATEGORY MATRIX

Consider a directed graph $G = (D, L)$ (the "document graph"), whose nodes $d \in D$ represent *documents* and whose arcs $(d, d') \in L$ represent *(hypertextual) links* between documents. Further assume that we have a set $C$ of *categories* and that each document $d \in D$ is assigned a set of categories $C_d \subseteq C$.

Our first goal is to reconstruct the most plausible latent "category matrix" that explains the observed document graph; more precisely, we wish to find a $C \times C$ real-valued matrix $W$ such that

$$\sum_{c \in C_d} \sum_{c' \in C_{d'}} w_{c,c'} \tag{1}$$

is positive iff $(d, d') \in L$.

We are going to assume that in most cases a relation is *unexpected* – that is, surprising to the reader – if it is poorly explained by a plausible category matrix. We will put this assumption under test in the experimental section.

To find such a matrix $W$, we recast our goal in the framework of online binary classification. Binary classification is a well-known problem in supervised machine learning. Suppose to have a traning set of *examples*, each one associated with a binary label $\widehat{y}_i \in \{-1, 1\}$; based on these data, the problem is to build a classifier able to label correctly unknown data. *Online* classification simplifies this problem by assuming each example is presented in a sequential fashion; the classifier (1) observes an example; (2) tries to predict its label; (3) receives the true label, and consequentially updates its internal state; (4) moves on to the next example. An online learning algorithm, generally, needs a constant amount of memory with respect to the number of examples, which allows to employ these algorithms in a situation where a very large set of voluminous input data is avaiable – like in our case.

A well-known type of online learning algorithms are the so-called perceptron-like algorithms. They all share these traits: each example must be a vector $\mathbf{x}_i \in \mathbb{R}^n$; the internal state of the classifier is also represented by a vector $\mathbf{w} \in \mathbb{R}^n$; the predicted label is $y_i = sign(\mathbf{w} \cdot \mathbf{x}_i)$, and the algorithms differ on how $\mathbf{w}$ is built. Perceptron-like algorithms (for example, ALMA and Passive-Aggressive) are usually simple to implement, provide tight theoretical bounds, and have been proved to be fast and accurate in practice [10, 8]. For these reasons, we will reduce our problem to online binary

classification.

To this aim, let us represent each document $d$ with the indicator vector of $C_d$, i.e., with the binary vector $\mathbf{d}$ such that $d_c = 1$ *iff* $c \in C_d$. Now, an *example* will be a pair of documents $(d, d')$, represented as the outer product kernel $\mathbf{d} \otimes \mathbf{d}'$: this is a matrix where the element $[\mathbf{d} \otimes \mathbf{d}']_{c,c'}$ is 1 iff the first document belongs to $c$ and the second to $c'$. This $(|C| \times |C|)$-matrix[1] can be alternatively thought of as a vector of size $|C|^2$, allowing us to use them as training examples for a perceptron-like classifier, where the label is $y = 1$ iff $(d, d') \in L$ (if there is a link), and $y = -1$ otherwise. The learned vector $\mathbf{w}$ will be, if seen as a $|C| \times |C|$ matrix, the desired $W$ appearing in (1). In other words, we are using $|C|^2$ features, in fact a kernel projection of a space of dimension $2|C|$ onto the larger space of size $|C|^2$. Similarly the weight vector to be learned has size $|C|^2$. Positive examples are those that correspond to existing links.

*A Passive-Aggressive algorithm.* Among the existing perceptron-like online classification frameworks, we chose the well-known Passive-Aggressive classifier, characterized by being extremely fast, simple to implement, and shown by many experiments [6, 17] to perform well on similar datasets. To cast this algorithm for our case, let us consider a sequence of pairs of documents

$$(d_1, d'_1), \ldots, (d_T, d'_T) \in D^2$$

(to be defined later). Define a sequence of matrices $W_0, \ldots, W_T$ and of slack variables $\xi_1, \ldots, \xi_T \geq 0$ as follows:

- $W_0 = 0$
- $W_{t+1}$ is a matrix minimizing $\|W_{t+1} - W_t\| + K\xi_{t+1}$ subject to the constraint that

$$\sigma(d_t, d'_t) \cdot \sum_{c \in C_{d_t}} \sum_{c' \in C_{d'_t}} w_{t+1}(c, c') \geq 1 - \xi_{t+1}, \qquad (2)$$

  where

$$\sigma(x, y) = \begin{cases} -1 & \text{if } (x, y) \notin L \\ 1 & \text{if } (x, y) \in L \end{cases},$$

  $\| - \|$ denotes the Frobenius norm and $K$ is an optimization parameter determining the amount of aggressiveness.

The intuition behind the above-described optimization problem [8] is the following:

- the left-hand-side of the inequality (2) is positive iff $W_{t+1}$ correctly predicts the presence/absence of the link $(d_t, d'_t)$; its absolute value can be thought of as the confidence of the prediction;
- we would like the confidence to be at least 1, but allow for some error (embodied in the slack variable $\xi_{t+1}$);
- the cost function of the optimization problem tries to keep as much memory of the previous optimization steps as possible (minimizing the difference with the previous iterate), and at the same time to minimize the error contained in the slack variable.

By merging the Passive-Aggressive solution to this problem with our aforementioned framework, we obtain the algorithm described in Alg. 1.

**Algorithm 1** Passive-Aggressive algorithm to build the latent category matrix.

INPUT:
Categories $C_d \subseteq C$ for each document $d \in D$
A sequence $(d_1, d'_1), \ldots, (d_T, d'_T)$ of elements of $D \times D$
A parameter $K > 0$
OUTPUT:
The latent category matrix $W$

1. $W \leftarrow \mathbf{0}$
2. For $i = 1, \ldots, T$
   (a) $\rho \leftarrow \frac{1}{|C_{d_i}| \cdot |C_{d'_i}|}$
   (b) $\mu \leftarrow \sum_{c \in C_{d_i}} \sum_{c' \in C_{d'_i}} W_{c,c'}$
   (c) **If** $(d_i, d'_i) \in L$
   $\delta \leftarrow \rho \cdot \min(K, 1 - \mu\rho)$
   **else**
   $\delta \leftarrow -\rho \cdot \min(K, 1 + \mu\rho)$
   (d) For each $c \in C_{d_i}$, $c' \in C_{d'_i}$:
   $W_{c,c'} \leftarrow W_{c,c'} + \delta$

Please note that our aim is not to build a perfect classifier: instead, we will use this algorithm to find a plausible category-category matrix. This can be seen as a rivisitation of the use of classifiers to detect outliers, as described for example in [2].

*Sequence of pairs.* In our case, $W$ is built through a single-pass online learning process, where we have all positive examples at our disposal (and they are in fact all included in the training sequence), but where negative examples cannot be all included, because they are too many and they would produce overfitting.

The Passive-Aggressive construction described above depends crucially on the sequence of positive and negative examples $(d_1, d'_1), \ldots, (d_T, d'_T)$ that is taken as input. In particular, as discussed in [12], it is critical that the number of negative

[1] In practice, we normalize this matrix so that it has unit $L1$-norm, both because this is a common practice in the perceptron-like algorithms and because documents belonging to few categories provide stronger signals than those that belong to many categories.

and positive examples in the sequence is balanced. Taking this suggestion into account, we build the sequence as follows: nodes are enumerated (in arbitrary order), and for each node $d \in D$, all arcs of the form $(d, -) \in E$ are put in the sequence, followed by an equal number of pairs of the form $(d, -) \notin E$ (for those pairs, the destination nodes are chosen uniformly at random). Of course, if $m = |E|$ is the number of links, then $T = 2m$ and the sequence contains all the $m$ links along with $m$ non-links.

Obviously, there are other possible ways to define the sequence of examples and to select the subset of negative examples. We suggest some of them in Section 7. However, we chose to adopt this technique – single pass on a balanced random sub-sample of pairs – in order to test our methodology with a single, natural and computationally efficient approach.[2]

# 4. A NAIVE WAY TO BUILD THE CATEGORY MATRIX

Let us describe an alternative, easier, naive variant of how the latent category matrix $W$ could be obtained. Recall that the purpose is to use equation (1) to compute the expectedness of a link $(d, d')$.

For a given category $c$, let $D_c$ be the set of documents that have the category $c$; let also $\mathcal{E}_{c,d}$ represent the event that $d$ belongs to the category $c$ (i.e., $c \in C_d$ or, equivalently, $d \in D_c$). Now for any two categories $c$ and $c'$ one can compute the probability that there is a link between two documents that belong to those categories as

$$p_{c,c'} = P[(d, d') \in L \mid \mathcal{E}_{c,d} \text{ and } \mathcal{E}_{c',d'}].$$

This quantity can be naively estimated as the fraction of pairs $(d, d')$ such that $\mathcal{E}_{c,d} \wedge \mathcal{E}_{c',d'}$ that happen to be links. In other words,

$$p_{c,c'} = \frac{|\{(D_c \times D_{c'}) \cap L\}|}{|D_c| \cdot |D_{c'}|}.$$

For a specific pair of documents $(d, d')$, the probability of the presence of a link is given by

$$P\left[(d, d') \in L \middle| \bigcap_{c \in C_d} \mathcal{E}_{c,d} \text{ and } \bigcap_{c' \in C_{d'}} \mathcal{E}_{c',d'}\right].$$

Now, under some independence assumptions[3], the latter can be expressed as

$$\prod_{c \in C_d} \prod_{c' \in C_{d'}} P\left[(d, d') \in L \middle| \mathcal{E}_{c,d} \text{ and } \mathcal{E}_{c',d'}\right] = \\ = \prod_{c \in C_d} \prod_{c' \in C_{d'}} p_{c,c'}.$$

[2]We carried out experiments performing more passes on the same subsample; it slightly increased (less than 2%) the *accuracy* of $W$ – i.e., the number of pairs that are correctly classified. However, it is dubious whether the increased time cost is worth the limited improvement in terms of unexpectedness mining.

[3]More precisely, we are assuming that $\mathcal{E}_{c,d}$ and $\mathcal{E}_{c',d'}$ are independent, whenever $c \neq c'$ or $d \neq d'$, and also that they are independent even under the knowledge that $(d, d') \in L$.

Applying a logarithm, this is rank-equivalent to

$$\sum_{c \in C_d} \sum_{c' \in C_{d'}} w_{c,c'}$$

where

$$w_{c,c'} = \log p_{c,c'} = \log \frac{|\{(D_c \times D_{c'}) \cap L\}|}{|D_c| \cdot |D_{c'}|}$$

This is yet another way to define the matrix $W$ used in the LlamaFur algorithm; the resulting expectedness score for link $(d, d')$ is given by (1), and will be referred to as Naive-LlamaFur.

# 5. USING THE CATEGORY MATRIX

Let us now call $W$ the category matrix obtained at the end of the learning process (that is, $W = W_T$, according to the notation of Section 3), or equivalently the matrix built using the naive approach of Section 4. This matrix allows one to sort the links $(d, d') \in L$ in increasing order of $\sum_{c \in C_{d_t}} \sum_{c' \in C_{d'_t}} w_{c,c'}$ (i.e., by increasing explainability): the first links are the most unexpected.

In particular, in the case of the learning approach of Section 3, one can build a graph $G^* = (D, L^*)$ whose links are the set $L^*$ of pairs $(d, d')$ such that

$$\sum_{c \in C_{d_t}} \sum_{c' \in C_{d'_t}} w_{c,c'} \geq 0.$$

In a standard binary-classification scenario, $G^*$ would be the graph $G$ that our classifier learned. In particular, the elements of the set $L \backslash L^*$ ($L^* \backslash L$, resp.) are the false negative (false positive, resp.) instances.

But ours is *not* a link-prediction task, and we do not expect in any sense that $L$ and $L^*$ are similar. In particular, we shall certainly observe a phenomenon that we can call *generalization effect*: suppose that it frequently happens that a document assigned to a category $c$ (e.g., an actor) contains links to documents assigned to another category $c'$ (e.g., a movie). This will probably make $w_{c,c'}$ very large, and so we may falsely deduce that *every* document assigned to $c$ (every actor) contains a link to *every* document assigned to $c'$ (every movie).

The generalization effect will, by itself, make $L^*$ much larger than $L$ (i.e., it will produce many false positive instances), but we do not care much about this aspect. Our focus is *not* on trying to reconstruct $L$, but rather in understanding which elements of $L$ are difficult to explain based on the categories of the involved documents. We say that a link $(d, d') \in L$ is *explainable* iff $(d, d') \in L^*$; the set of explainable links is therefore $L \cap L^*$. On the contrary, the elements of $L \setminus L^*$ are called unexplainable, and these are the links we want to focus on.

In Figure 1 we show two small examples of how the matrix $W$ learned as in Section 3 looks like, when considering the Wikipedia dataset (for a full explanation of how the dataset was built, see Section 6): in the picture, we display the 18 neighbours closer to two starting categories ("Science Fiction Films" and "Keyboardists"); the width of the arc from $c$ to

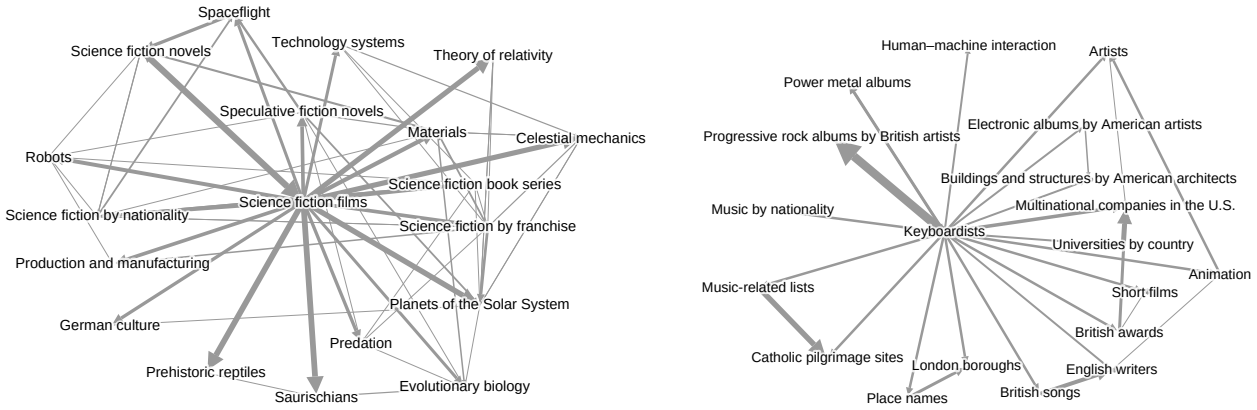



**Figure 1: Two fragments of the latent category graph induced by LlamaFur matrix $W$, representing the 18 closer neighbors of categories "Science Fiction Films" and "Keyboardists", respectively. The width of the arc from $c$ to $c'$ is proportional to $w_{c,c'}$, and arcs with $w_{c,c'} \leq 1$ are not shown.**

$c'$ is proportional to $w_{c,c'}$, and arcs with $w_{c,c'} \leq 1$ are not shown.

For example, from the picture it is clear that a link from a page of a science-fiction film to a page of a science-fiction novel is highly expected, as it is one from a page of a keyboardist to one of a british progressive rock album.

The rougher version induced by Naive-LlamaFur is shown in Figure 2.

# 6. EXPERIMENTS

Given its increasing importance in knowledge representation [20], we used the English edition of Wikipedia as our testbed. In praticular, we employed the `enwiki` snapshot[4] of February 3, 2014 to obtain:

- the document graph, composed by 4 514 662 Wikipedia pages, with 110 699 703 arcs; every redirect was merged to its target page;
- the full categorization of pages: a map associating every page to one of the 1 134 715 categories;
- the category pseudo-tree: a graph built by Wikipedia editors, with the aim of assigning each category to a "parent" category.

*Wikipedia categories.* The first problem is that the categorization on Wikipedia is quite noisy and, in fact, a continuous work-in-progress: therefore, categories may contain only one (or even no) page, they might be duplicates of each other, and so on. The obvious solution would be to use the pseudo-tree to find the top categories; but the category tree is a work-in-progress itself. Not only – as stated[5] by Wikipedia – "categories can be sub-categories of themselves", but cycles are also present: for example, the largest strongly connected component has 6 833 categories, all direct or indirect subcategories of one another.

We therefore cleansed the page categorization as follows: we computed the harmonic centrality measure [4] on the category pseudo-tree, and considered only the set $C$ of the 20 000 most central categories. To give an idea about the effectiveness of this simple method in capturing the generality of categories, we report in Table 1 the first and the last categories on our list[6].

We then computed, for every category $c$, the category $\iota(c) \in C$ closest to $c$ in the pseudo-tree, and re-categorized all the pages applying $\iota(-)$ to its original categories. If there is no $c' \in C$ connected to $c$, $\iota(c)$ is undefined and we simply discarded $c$. In Table 2 we show some examples of this re-categorization of pages.

In the end, we obtained a set $C$ of 20 000 categories, and a map associating each Wikipedia page $d$ to $C_d \subset C$; on average, each page belongs to 4 categories. We procedeed then to apply LlamaFur to extract the latent category matrix $W$; the ratio $|L \cap L^*|/|L|$ – that is, how many existing links are explained by $W$ – is equal to 86%. We illustrated previously in Fig. 1 some fragments of $W$. Finally, we proceeded to assign our unexpectedness score to each link.

*Evaluation methodology.* We want to evaluate the effectiveness of LlamaFur using the standard framework commonly adopted in Information Retrieval. In our context, a *query* is a document, the possible *results* are the hyperlinks that the document contains, and a result is *relevant* for our problem if it represents an unexpected link. The scenario we have in mind is that of a user wishing to find surprising links in a certain Wikipedia page.

[4]This dataset is commonly referred to as `enwiki-20140203-pages-articles` according to Wikipedia naming scheme.

[5]See *"Known issues"* on `en.wikipedia.org/wiki/Wikipedia:FAQ/Categorization`.

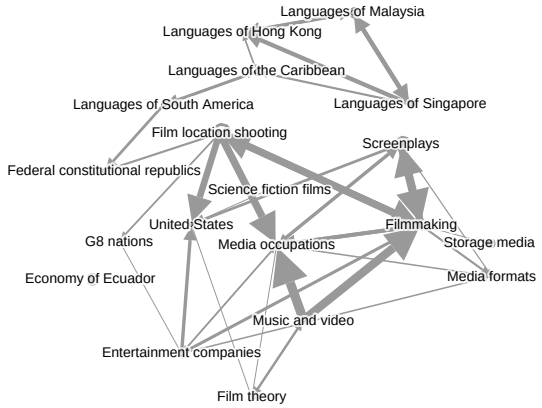
[6]We also excluded utility categories, like *"Categories by country"* and *"Main topic classifications"* – originally highly ranked.

| Rank | Category | Rank | Category |
|---|---|---|---|
| 1 | Countries | 19981 | Maldives |
| 2 | Society | 19982 | Government buildings on the National Register of Historic Places |
| 3 | Nationality | 19983 | Illinois waterways |
| 4 | Political geography | 19984 | Bodies of water of Illinois |
| 5 | Culture | 19985 | 2002 in association football |
| 6 | Humans | 19986 | Electronica albums by British artists |
| 7 | Social sciences | 19987 | Visitor attractions in Arkansas by county |
| 8 | Structure | 19988 | Years of the 20th century in Europe |
| 9 | Human–geographic territorial entities | 19989 | Commonwealth Games events |
| 10 | Contents | 19990 | Albums by English artists by genre |
| 11 | Geographic taxonomies | 19991 | American football in Pennsylvania |
| 12 | Fields of history | 19992 | Ethnic groups in Poland |
| 13 | Places | 19993 | Card games |
| 14 | Humanities | 19994 | Central African people |
| 15 | Continents | 19995 | Deaths by period |
| 16 | Political concepts | 19996 | Visitor attractions in Vermont |
| 17 | Human geography | 19997 | Ancient roads and tracks |
| 18 | Subfields of political science | 19998 | People in finance by nationality |
| 19 | Articles | 19999 | Populated places in Greater St. Louis |
| 20 | Subfields by academic discipline | 20000 | Religion in Poland |

**Table 1: Topmost and bottommost wikipedia categories according to their harmonic centrality in the Wikipedia category graph.**

| **Original category** $c$ | **Substitution** $\iota(c) \in C$ |
|---|---|
| Southern Tang poets | Poets by nationality |
| Antsiranana Province | Country subdivisions of Africa |
| Fellows of Magdalen College, Oxford | University of Oxford |
| Actresses from Greater Manchester | Greater Manchester |
| Guyanese slaves | History of South America |
| Swiss manuscripts | Swiss culture |
| Wilson Pickett songs | Songs by artist |
| Baroque architecture in Austria | Baroque architecture by country |
| Eastern Collegiate Roller Hockey Association | ∄ |
| Art schools in Washington (state) | Washington (state) culture |
| Rivers of Kostroma Oblast | Rivers by country |
| Flamenco compositions | Spanish music |
| Oil fields of Gabon | Geology of Africa |
| Basketball teams in Georgia (U.S. state) | Basketball teams in the United States by state |
| 2004 in Australian motorsport | 2004 in sports |
| Populated places established in 1821 | ∄ |
| Elections in Southwark | Local government in London |
| Permanent Representatives of Norway to NATO | Ambassadors of Norway |
| Basketball in Turkey | Basketball by country |
| Balli Kombëtar | ∄ |

**Table 2: An excerpt of the re-categorization process. We write ∄ if there is no category in $C$ connected to $c$.**



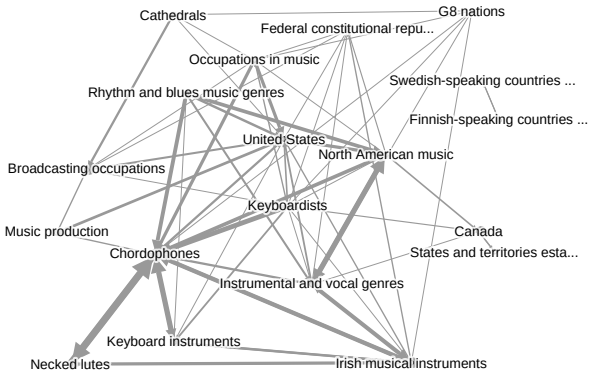




**Figure 2: Two fragments of the latent category graph induced by Naive-LlamaFur matrix, representing the 18 closer neighbors of categories "Science Fiction Films" and "Keyboardists", respectively. The width of the arc from $c$ to $c'$ is proportional to $w_{c,c'}$, and the lighter arcs are not shown. For comparison with LlamaFur, see Figure 1.**

| Label | Fraction |
|---|---|
| Totally Unexpected | 2.3% |
| Unexpected | 8.9% |
| Expected | 30.8% |
| Totally Expected | 58.0% |

**Table 3: Distribution of labels obtained from human evaluation.**

In order to compare the results obtained by LlamaFur with the existing state-of-the-art for similar problems, we performed a user study based on the same pooling method adopted for many standard collections such as TREC (`trec.nist.gov`): we considered a random sample of 237 queries (i.e., Wikipedia documents); for each query we took, among its $t$ possible results (i.e., links), the top-$\lfloor \alpha \cdot t \rfloor$ most unexpected ones according to each system under comparison (see below); all the resulting links were evaluated by human beings. We set $\alpha = 0.1$, and obtained about 3 620 links.

The human evaluators were asked to categorize each link into one of four classes ("totally expected", "slightly expected", "slightly unexpected" and "totally unexpected"). After the human evaluation, we only considered the queries that have at least one irrelevant ("totally/slightly expected") and one relevant ("totally/slightly unexpected") result according to the evaluation, obtaining a dataset with 117 queries. In this dataset, on average each query has 3.45 relevant results over 20.56 evaluated links. The distribution of labels is reported in Table 3.

*Baselines and competitors.* In our comparison, LlamaFur is tested in combination and against a number of baselines and competitors. In particular, we considereded LlamaFur and its naive variant, Naive-LlamaFur, along with some of the other (un)expectedness measures proposed in the literature.

Albeit there are, at the best of our knowledge, no algorithms specifically devoted to determining unexpected links, we can adapt some techniques used for unexpected documents to our case. All of those methods try to measure the unexpectedness of a document $d$ among a set of retrieved documents $R$. In our application, we are considering a link $(d', d)$ and taking $R$ to be the set of all documents towards which $d'$ has a hyperlink.

- *Text-based methods.* In the literature, all of the measures of unexpectedness are based on the textual content of the document under consideration.
  - The first index, called M2 in [11] (a better variant of $M1$, the measure proposed in [14]), is defined as:
    $$\mathrm{M2}(d) = \frac{\sum_t U(d, t, R)}{m}$$
    where $m$ is the number of terms in the dictionary, and $U(d, t, R)$ is the maximum between 0 and the difference between the normalized term frequency of term $t$ in document $d$ and the normalized term frequency of $t$ in $R$ (the set of all retrieved documents). The normalized term frequency is the frequency of a term divided by the frequency of the most frequent term.
  - The second index, called M4 in [11] (where they prove that it works better than M2 in their context), is the
    $$\mathrm{M4}(d) = \max_t \mathrm{tf}(t) \cdot \log \frac{|R|}{\mathrm{df}(t)}$$
    where $\mathrm{tf}(t)$ is the normalized term frequency of term $t$ in $d$, and $\mathrm{df}(t)$ the number of documents in $R$ where $t$ appears.
- *Link-prediction methods.* A completely different, alternative approach to the problem is based on *link prediction*: how likely is it that the link $(d', d)$ is created, if we assume that it is not there? Among the many

techniques for link prediction [15], we tested the well-known *Adamic-Adar index* [1] (AA, in the following), defined[7] by

$$\mathrm{AA}(d, d') = \sum_{d'' \in \Gamma(d) \cap \Gamma(d')} \frac{1}{\log |\Gamma(d'')|},$$

where $\Gamma(d)$ is the set of documents which $d$ links to.

- *Combinations.* Besides testing all the described techniques in isolation, we tried to combine them linearly. Since each unexpectedness measure exhibits a different scale, we first need to normalize each measure by taking its *studentized residual*[8] [7].

***Results.*** In the following, we are only going to discuss the best algorithms and combinations, besides some of the most interesting alternatives. The raw average bpref [5] values are displayed in Table 4. Figure 4 shows, for each algorithm, how many queries have obtained a certain bpref value; for the sake of readability, we have grouped bpref values into four groups. Ideally, an algorithm should produce as many large bpref values as possible. LlamaFur is the one single algorithm that goes closer to the target, whereas M4 and AA are the second best with a small margin. Naive-LlamaFur is worse, wheras M2 is the overall worst. Interestingly, combining pure link prediction methods (AA) with LlamaFur significantly improves AA of about 28%.

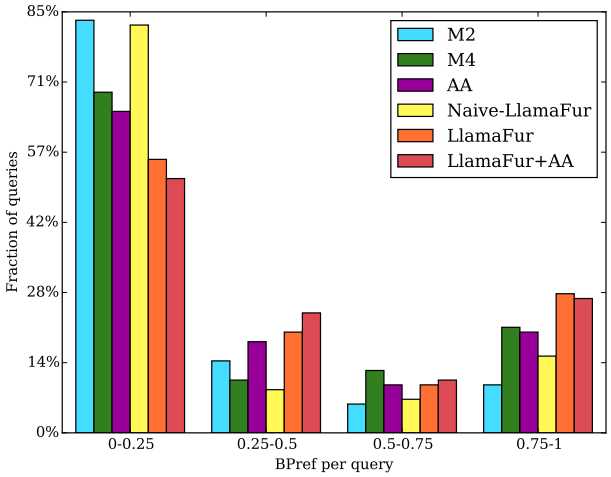



**Figure 4: Distribution of the bpref measure over each evaluated query. Average values are shown in Table 4.**

Some complementary information about the behaviour is provided by the precision-recall graph of Figure 3: first of all, LlamaFur, AA, M4 and their combinations have larger precision than the remaining ones for almost all the recall levels; on the other hand LlamaFur +AA is the best method for recall values up to 50%, and LlamaFur has definitely better precision than AA until 30% of recall.

In fact, M4, AA and LlamaFur seem to be complementary to one another; in some sense, this is not surprising given that they stem from completely different sources of information: one is based on the textual content, another on the pure link graph and the latter on the category data.

Some further clue on the behaviour of LlamaFur is provided by Figures 5 and 6, where the distribution of LlamaFur and LlamaFur + AA expectedness values is shown for each of the four labels provided by the human evaluation. The red line is the median.

[7]The formula is applied to the symmetric version of the graph, in our case; note that this (like LlamaFur) is a measure of expectedness, whereas M2 and M4 are measures of unexpectedness.

[8]The (internally) studentized residual is obtained by dividing the residual (i.e., the difference from the sample mean) by the sample standard deviation.

| Algorithm | Average bpref | Input data |
|---|---|---|
| AA | 0.288 | graph |
| M2 | 0.179 | bag of words |
| M4 | 0.290 | bag of words |
| Naive-LlamaFur | 0.216 | graph, categories |
| LlamaFur | 0.364 | graph, categories |
| LlamaFur + AA | 0.372 | graph, categories |

**Table 4: Average values for bpref.**

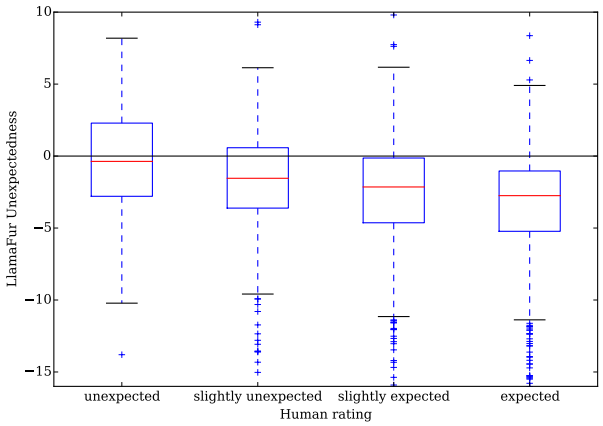



**Figure 5: Comparison of the unexpectedness evaluated by LlamaFur with equation (1) over the different labels obtained from human evaluation.**

## 7. CONCLUSIONS AND FUTURE WORK

In this work we presented a technique to find unexpected links in hyperlinked document corpora based on the determination of a latent category matrix that explains common links; the latter is built using a perceptron-like technique. We show that our method provides better accuracy than most existing text-based techniques, with higher efficiency and relying on a much smaller amount of information. An interesting question is whether the latent category matrix can be used to improve link prediction *per se*, i.e. if it is useful to find links and not only unexpected ones: this problem

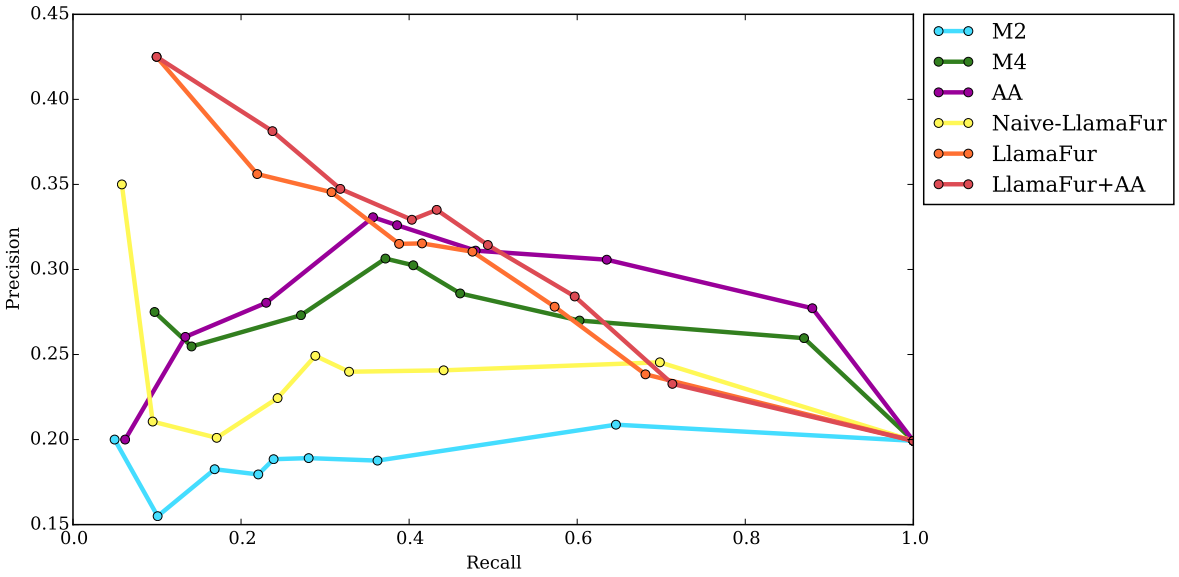



**Figure 3: Average precision-recall values evaluated after the 1st, 2nd, 5th, 8th, 10th, 15th, 25th, 50th, and 100th percentiles for each query.**

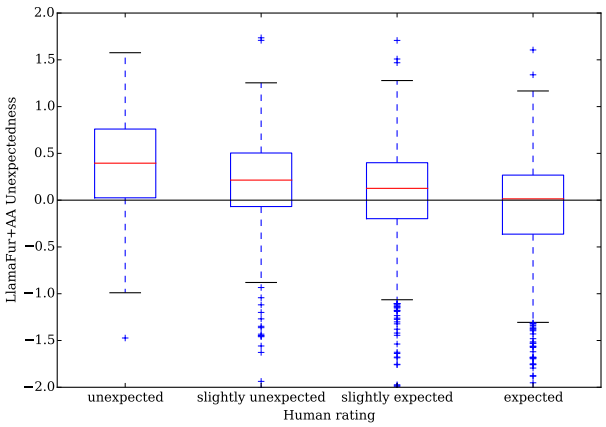



**Figure 6: Comparison of the unexpectedness score evaluated by LlamaFur + AA over the different labels obtained from human evaluation.**

requires that one finds a way to bypass the generalization effect that the matrix produces.

Another possible direction would be to try different approach to the classification problem described in Section 3, in order to improve its effectiveness. To this aim, one could recast the problem as a cost-sensitive classification where false negatives are more costly than false positives. Other useful techniques include active learning [16]: since we need a subset of the non-linked pairs as counter-examples, active learning would select the more effective ones. An alternative approach to the same task would be to employ one-class learning [13]. This is left as future work.

# 8. REFERENCES


[1] Lada A. Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25:211–230, 2001.

[2] Charu C Aggarwal. *Outlier analysis*. Springer Science & Business Media, 2013.

[3] Malik Agyemang, Ken Barker, and Reda Alhajj. Hybrid approach to web content outlier mining without query vector. In A. Min Tjoa and Juan Trujillo, editors, *DaWaK*, volume 3589 of *Lecture Notes in Computer Science*, pages 285–294. Springer, 2005.

[4] Paolo Boldi and Sebastiano Vigna. Axioms for centrality. *Internet Mathematics*, 10(3-4):222–262, 2014.

[5] Chris Buckley and Ellen M Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32. ACM, 2004.

[6] Vitor R Carvalho and William W Cohen. Single-pass online learning: Performance, voting schemes and online feature selection. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 548–553. ACM, 2006.

[7] R. Dennis Cook and Sanford Weisberg. *Residuals and*

*Influence in Regression*. Monographs on Statistics and Applied Probability, 18. Chapman and Hall/CRC, 1983.

[8] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, 2006.

[9] Damien Eklou, Yasuhito Asano, and Masatoshi Yoshikawa. How the web can help wikipedia: A study on information complementation of wikipedia by the web. In *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, ICUIMC '12, pages 9:1–9:10. ACM, 2012.

[10] Claudio Gentile. A new approximate maximal margin classification algorithm. *J. Mach. Learn. Res.*, 2:213–242, 2002.

[11] François Jacquenet and Christine Largeron. Discovering unexpected documents in corpora. *Knowledge-Based Systems*, 22(6):421 – 429, 2009.

[12] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[13] Shehroz S Khan and Michael G Madden. A survey of recent trends in one class classification. In *Artificial Intelligence and Cognitive Science*, pages 188–197. Springer, 2010.

[14] Bing Liu, Yiming Ma, and Philip S. Yu. Discovering unexpected information from your competitors' web sites. In Doheon Lee, Mario Schkolnick, Foster J. Provost, and Ramakrishnan Srikant, editors, *KDD*, pages 144–153. ACM, 2001.

[15] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150 – 1170, 2011.

[16] Edwin Lughofer. Single-pass active learning with conflict and ignorance. *Evolving Systems*, 3(4):251–271, 2012.

[17] Corrado Monti, Alessandro Rozza, Giovanni Zappella, Matteo Zignani, Adam Arvidsson, and Elanor Colleoni. Modelling political disaffection from twitter data. In *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*, page 3. ACM, 2013.

[18] Akiyo Nadamoto, Eiji Aramaki, Takeshi Abekawa, and Yohei Murakami. Content hole search in community-type content. In *Proceedings of the 18th international conference on World wide web*, pages 1223–1224. Association for Computing Machinery, 2009.

[19] S.P. Ponzetto and M. Strube. Deriving a large scale taxonomy from wikipedia. *Proceedings of the National Conference on Artificial Intelligence*, 2:1440–1445, 2007.

[20] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3):203 – 217, 2008. World Wide Web Conference 2007Semantic Web Track.

[21] Kosetsu Tsukuda, Hiroaki Ohshima, Mitsuo Yamamoto, Hirotoshi Iwasaki, and Katsumi Tanaka. Discovering unexpected information on the basis of popularity/unpopularity analysis of coordinate objects and their relationships. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 878–885. ACM, 2013.