Why Inverse Document Frequency?

Kishore Papineni
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598, USA
papineni@us.ibm.com

Abstract

Inverse Document Frequency (IDF) is a popular measure of a word's importance. The IDF invariably appears in a host of heuristic measures used in information retrieval. However, so far the IDF has itself been a heuristic. In this paper, we show IDF to be optimal in a principled sense. We show that IDF is the optimal weight of a word with respect to minimization of a Kullback-Leibler distance suitably generalized to nonnegative functions which need not be probability distributions. This optimization problem is closely related to maximum entropy problem. We show that the IDF is the optimal weight associated with a word-feature in an information retrieval setting where we treat each document as the query that retrieves itself. That is, IDF is optimal for document self-retrieval.

1 Introduction

Inverse Document Frequency (IDF) is a popular measure of a word's importance. It is defined as the logarithm of the ratio of number of documents in a collection to the number of documents containing the given word (Sparck Jones, 1973). This means rare words have high IDF and common function words like "the" have low IDF. IDF is believed to measure a word's ability to discriminate between documents. IDF invariably appears in a host of heuristic measures used in information retrieval (Salton and McGill, 1983). However, so far IDF has itself been a heuristic, although a good one.

In this paper, we show IDF to be optimal in a precise sense. To show this, we first view information retrieval as a classification problem with each document in the collection being a class. We then build a classifier that scores the documents given a query. To train this classifier, we treat each document as a query that retrieves itself. Our classifier is an exponential model similar to the one in the maximum entropy framework, but without the usual normalization. Therefore, our classifier does not produce a conditional probability distribution. In the regular maximum entropy framework, one chooses weights in the exponential model to maximize the likelihood of the training data. However, we must now maximize a *relaxed* likelihood since our classifier uses a generalized distribution. Our relaxed likelihood is globally convex just like the traditional likelihood. In the case when there is a single binary feature in the model, the optimal solution is stunningly simple in contrast to the solution in the regular conditional maximum entropy framework. A word-feature that examines the occurrence of the word in both the query and the document is a binary feature. IDF is the optimal weight of this feature. That is, *IDF is optimal for document self-retrieval.*

In the exponential modeling framework, the weight associated with a feature is not the same as the "importance" or gain associated with the feature. The gain is the improvement of the relaxed likelihood of the model (prior plus the single feature) over that of the prior. In our self-retrieval setting with a fixed collection of documents, the least important words are those with a single count or those occurring in almost all the documents.

IDF is the optimal weight when there is a single feature in the model. But how do we share the word's weight across a unigram and a bigram? The traditional justifications of IDF and similar measures assume that words occur independently in documents, e.g. (Robertson and Sparck Jones, 1976). Those approaches do not easily generalize to account for word co-occurrences. Our method does. It can treat simultaneous features that examine presence or absence of single words, phrases, and more sophisticated linguistic relations such as synonyms and hyponyms. With the independence assumption, our approach indeed produces the IDF. In the more general case, we iteratively solve the optimization problem with word co-occurrences. Another approach that does not make the independence assumption is in (Wong and Yao, 1992). They show that IDF of a word is the complement of relative entropy, upto scaling by a constant independent of the word, if we distribute the probability mass equally among documents containing the word. Their framework does not decouple the weight of the word from its goodness as evidence.

For the interesting question of sharing a word's weight across a unigram and a bigram, we have an explicit closed-form solution. The unigram and bigram are overlapping features. With a simple linear transformation, they become non-overlapping. It turns out that the multi-parameter optimization for non-overlapping features is reduced to the individual single parameter problems on the component features. That is, with non-overlapping features the problem decouples into many simple problems. We recover the solution to the problem with overlapping features from the easily computed closed-form solution to that with non-overlapping features.

Since importance of a feature is given not by its IDF but by the improvement in likelihood from that feature, we can examine when a bigram becomes important when the prior already has the two component word features. The bigram feature is important only when it improves the retrieval likelihood over the component word-features. We show a ranking of bigrams on Reuters-21578 corpus with respect to their generalized likelihood improvement.

2 Retrieval as Classification

We treat information retrieval as a classification problem with each document in the collection as a class. To build a classifier, we are given a set of labeled training data as (query, document) pairs.

A traditional method of classification employs a probability distribution on the classes conditioned on the observation (query), for example, using a decision tree. For any observation, we assign nonnegative scores (probabilities) to the classes based on the trained model. Of course, some applications like information retrieval do not require that the model be a true probability distribution, i.e., that the scores add up to one; a probability distribution on the documents is not necessary.

Our framework is similar to the exponential models constructed in terms of features and their weights in the familiar conditional maximum entropy (minimum Kullback-Leibler "distance") framework (Della Pietra et al., 1997). We discuss this framework below.

The standard exponential model has three components: a prior conditional probability distribution $P_0(c|x)$; a vector $\phi(x,c)$ of feature functions (questions or rules) on observation and the candidate class; and a weights vector λ corresponding component-wise to the feature vector. The prior distribution embodies prior knowledge about the domain, if any. If the modeler does not have any prior knowledge, $P_0(c|x)$ can simply be the uniform distribution. The features are usually, but not required to be, binary. The features are questions on the observation as well as the candidate class, in contrast to traditional decision tree questions. An example is a binary-valued feature that examines if a particular word occurs in both the query and the candidate document. Another example is a binary feature that examines if the query contains a particular word and the document contains a synonym or a hyponym of the word.

The three components are combined to form an exponential model as below:

$$P(c|x) := \frac{P_0(c|x)e^{\lambda\cdot\phi(x,c)}}{Z_x}$$

where

$$Z_x := \sum_{c\in\mathcal{C}} P_0(c|x)e^{\lambda\cdot\phi(x,c)}$$

is the normalization term needed to make P a probability distribution. $\mathcal{C}$ above is the set all of classes, assumed to be finite and fixed *a priori*. Recall that λ and ϕ are vectors and therefore

$$\lambda\cdot\phi(x,c) = \sum_i \lambda_i\phi_i(x,c).$$

With $\alpha_i := e^{\lambda_i}$, we have

$$P(c|x) := \frac{P_0(c|x)\prod_i \alpha_i^{\phi_i(x,c)}}{Z_x}.$$

Therefore, the model P combines the information from the training data and the prior model. We can view the evaluation of the probabilities as a voting scheme by experts. Each feature is an "expert" with a "vote" (α_i). An expert votes only when he agrees with the observation and the candidate class. The votes are multiplied and are followed by a normalization.

The model is specified by the functional form, training data to tune the parameters on, and an objective function to tune the parameters. Maximum likelihood is a classic objective function and is equivalent to maximum entropy in the special case of uniform prior distribution. In the general prior case, maximum likelihood is equivalent to minimum Kullback-Leibler "distance" subject to expectation constraints on the features and is globally convex.

The standard exponential model above contains more components than we need. As we argued before, the normalization by Z_x is not necessary in many applications like information retrieval. If we drop the normalization factor, we must formulate a well-defined optimization problem on the new unnormalized exponential models. Traditional likelihood is not well-defined on nonnegative functions. It turns out that there is an appropriate generalization to entropy maximization in the unnormalized case that also results in a globally convex objective function. An added benefit is that the generalization results in a closed form solution to the

single-parameter binary feature problem. Remarkably, that closed-form solution is nothing other than the IDF of a word for a feature that examines the occurrence of the word both in the query and the document.

First, we establish the parallel notation for nonnegative functions. We denote the scoring functions $Q : (x, c) \mapsto [0, \infty)$ as $Q(c|x)$ rather than as $Q(x, c)$ to make the parallel explicit. Suppose the training data is given as $\{(x_i, c_i)\},\ i = 1, 2, \cdots, N$.

For any function g on the training data, we define an analog of expectation:

$$Q[g] := \sum_{i=1}^{N} \sum_{c \in \mathcal{C}} Q(c|x_i) g(x_i, c).$$

We start with a prior nonnegative function $Q_0(c|x)$ that is not necessarily a probability distribution. We fix a feature vector $\phi(x, c)$. We define a parametric unnormalized convex exponential family of functions as below:

$$\mathcal{Q} \ := \ \{Q(c|x) := Q_0(c|x) e^{\lambda \cdot \phi(x,c)}\}$$

The models are log-linear in the parameters. In order to define the new objective function, we recall the notion of log likelihood of training data according to a conditional *probability distribution* $P(c|x)$ as below:

$$\sum_{i=1}^{N} \log P(c_i|x_i) \ .$$

In extending the log likelihood to a nonnegative function Q, it is not enough to simply replace P by Q in the above summation: for then it would be possible to increase the objective function indefinitely by simply scaling any arbitrary model without exploring much of the model space. It is therefore necessary to *balance* such scaling effects in the objective function. We use a simple balancing for our generalization, below.

We now define a generalized likelihood of training data according to Q as below, with $\mathbf{1}$ denoting the constant function:

$$G(Q) \ := \ N - Q[\mathbf{1}] + \sum_{i=1}^{N} \log Q(c_i|x_i) \ .$$

When Q is a probability distribution, $G(Q)$ clearly coincides with the traditional likelihood. The generalized likelihood is well-behaved with respect to scaling: Suppose we scale Q "row-by-row" for each x_i. Simple differentiation shows that optimal scaling results in a probability distribution. Thus it is not possible to increase the objective function indefinitely by simply scaling a model.

We now have a family of models and an objective function on them. It remains to optimize the objective function on the family. We first denote the empirical feature counts by d as below.

$$d \ := \ \sum_{i=1}^{N} \phi(x_i, c_i)$$

It is then routine to show that

$$G(Q) = G(Q_0) + Q_0[\mathbf{1}] - N + N + \lambda \cdot d - Q_0[e^{\lambda \cdot \phi}].$$

Defining

$$L(\lambda) := N + \lambda \cdot d - \sum_i \sum_c Q_0(c|x_i) e^{\lambda \cdot \phi(x_i, c)},$$

we see that

$$\sup_{Q \in \mathcal{Q}} G(Q)$$

is equivalent to

$$\sup_{\lambda} L(\lambda)$$

The smooth function $L(\cdot)$ is particularly tractable:

Observation 1. $L(\lambda)$ is globally strictly convex.

Analogous to the regular maximum entropy framework, the feature expectation with respect to the optimal solution matches the empirical counts:

Observation 2. The following is necessary and sufficient for a $\lambda_* \in R^n$ to be optimal:

$$d = \sum_i \sum_c Q_0(c|x_i) e^{\lambda_* \cdot \phi(x_i, c)} \phi(x_i, c)$$

In the next section, we present the problem of maximizing L as the dual of a generalized Kullback-Leibler distance minimization problem. In section 4, we solve the single-parameter optimization problem explicitly when ϕ is a scalar binary feature. In Section 5, we present a closed-form solution to the multi-parameter problem with non-overlapping features and use it compute the IDF's for a bigram and unigram that share a word. In Section 6, we present an analog of the TF-IDF formula used in information retrieval.

3 A Generalization of Kullback-Leibler Distance

Observation 2 shows that the G-optimal solution from the unnormalized exponential model class matches the empirical counts in expectation. Instead of searching over the exponential model class, can we search over all nonnegative functions that match the empirical counts in expectation optimizing some objective function and arrive at the same solution? That is the question we examine in this section. Given the training data as in Section 2,

the following modification of Kullback-Leibler "distance" between two conditional *probability distributions* is common in statistical natural language processing, e.g. (Lau et al., 1993). Assume $P_1(c|x) = 0$ whenever $P_2(c|x) = 0$.

$$D_{KL}(P_1, P_2) \ := \ \sum_{i=1}^{N} \sum_{c \in \mathcal{C}} P_1(c|x_i) \log \frac{P_1(c|x_i)}{P_2(c|x_i)} \ .$$

We consider the following generalization to *nonnegative* functions, $Q_1, Q_2 \geq 0$.

$$D_G(Q_1, Q_2) \quad := \quad N \ - \ Q_1[\mathbf{1}] \ + \ \sum_{i=1}^{N} \sum_{c \in \mathcal{C}} Q_1(c|x_i) \log \frac{Q_1(c|x_i)}{Q_2(c|x_i)}.$$

Clearly, if P_1 is a probability distribution, then $D_G(P_1, Q_2) = D_{KL}(P_1, Q_2)$.

This is one of many possible generalizations of the Kullback-Leibler distance. The Kullback-Leibler distance is itself a Bregman distance (Lafferty et al., 1997). However, our generalization is not a Bregman distance and hence differs fundamentally from the seemingly similar I-divergence considered in (Csiszar, 1991) (the same as the unnormalized relative entropy in (Collins et al., 2000)) even as it is identical to the latter when Q_2 is a probability distribution.

Recall the empirical feature counts given by

$$d \ := \ \sum_{i=1}^{N} \phi(x_i, c_i).$$

A priori, fix a $Q_0(c|x) \geq 0$, called the prior model. Define a family $\mathcal{R}$ as below:

$$\mathcal{R} \ := \ \{Q \geq 0 : Q[\phi] = d\}$$

We now pose an optimization problem:

$$\inf_{Q \in \mathcal{R}} D_G(Q, Q_0)$$

Formulating the Lagrangian

$$L(Q, \lambda) := D_G(Q, Q_0) + \lambda \cdot (d - Q[\phi]) \, ,$$

setting its differential with respect to $Q(\cdot|\cdot)$ to 0, we conclude that $Q(c|x)$ is of the form:

$$Q(c|x) = Q_0(c|x) e^{\lambda \cdot \phi(x,c)}.$$

Substituting this structural form into $L(Q, \lambda)$, we get a function of λ alone:

$$L(\lambda) = N + \lambda \cdot d - \sum_i \sum_c Q_0(c|x_i) e^{\lambda \cdot \phi(x_i, c)}.$$

Therefore, the primal problem is equivalent to the following dual:

$$\sup_{\lambda \in R^n} L(\lambda)$$

which was shown in Section 2 to be equivalent to

$$\sup_{Q \in \mathcal{Q}} G(Q).$$

Analogous to the regular maximum entropy framework (Della Pietra et al., 1997), it turns out that $\mathcal{R} \cap \bar{\mathcal{Q}}$ is a singleton containing the optimal solution. However, unlike in the regular maximum entropy framework, the optimal solution does not satisfy the Pythagorean property. The optimal vector solution is obtained by a variant of the improved iterative scaling algorithm (Papineni, 2000). However, in some special cases, the optimal solution has a closed-form solution to which we now turn.

4 The Scalar Binary Feature Problem

The scalar feature problem is important since we select good features from a pool of possibly millions of features. Feature selection involves solving a scalar feature problem for every feature in the pool for the optimal gain associated with the feature relative to the current model. Then the features are ranked by their gains and the top k features are selected and added to the current model. The model is retrained in a multi-parameter setting and the process is repeated until the optimal gain of the best feature in the remaining pool falls below a threshold. With the generalized Kullback-Leibler distance, the solution to the scalar binary feature problem turns out to be closed-form, unlike that in the regular conditional entropy maximization. Since feature is scalar, the empirical count d is now scalar. We define a scalar

$$d_0 := \sum_{i=1}^{N} \sum_{c \in \mathcal{C}} Q_0(c|x_i) \phi(x_i, c).$$

From Observation 2 and the fact that $e^{\lambda \phi} \phi = e^{\lambda} \phi$ for a scalar binary ϕ, we get the following.

Proposition 1. The binary scalar feature solution is given by

$$\lambda_* = \log \frac{d}{d_0},$$

with optimal gain

$$L(\lambda_*) - L(0) = d(\frac{d_0}{d} - 1 - \log \frac{d_0}{d}).$$

There is no such closed-form solution for a general prior and a general binary feature in the regular (normalized) *conditional* maximum entropy framework.

We now relate the optimal weight to the IDF of a word w. Identify x with a query and c with a document. Suppose $\mathcal{C}$ is the collection of documents $c_1, c_2, \cdots, c_N$. Suppose we define the training data as $\{(c_i, c_i)\}$, $i = 1, 2, \cdots, N$. Suppose we assign uniform prior distribution on the documents. That is, $Q_0(c|x) = Q_0(c) = \frac{1}{N}$. Define a scalar binary feature as below:

$$\phi_w(x, c) = \begin{cases} 1 & \text{if } w \text{ is in both } x \text{ and } c \\ 0 & \text{else.} \end{cases}$$

This feature has the property that we can split the binary question into two independent binary questions:

$$\phi_w(x, c) = \phi_w^1(x)\phi_w^2(c),$$

one question on the observation and one on the proposed class. We call a feature that admits such separation a *separable* feature. Then,

$$d := \sum_{i=1}^{N} \phi_w(c_i, c_i) = N_w$$

where N_w is the number of documents containing the word w. We also have

$$d_0 := \sum_{i=1}^{N} \sum_{c \in \mathcal{C}} \frac{1}{N} \phi_w^1(c_i)\phi_w^2(c)$$

$$= \frac{1}{N} \left(\sum_{i=1}^{N} \phi_w^1(c_i) \right) \left(\sum_{c \in \mathcal{C}} \phi_w^2(c) \right) = \frac{1}{N} N_w N_w.$$

Thus, we have

$$\lambda_* = \log \frac{N}{N_w} = \text{IDF}(w).$$

IDF is supposed to measure a word's importance. However, in our framework it is the optimal weight of a single word feature but is not the same as the importance of the word. We view the optimal gain associated with the feature as the word's importance. The gain is an information theoretic measure of goodness of a feature with respect to the prior and the training data. The gain *per document* for the above word feature is as follows:

$$\text{Gain} = \frac{N_w}{N} * (\frac{N_w}{N} - 1 - \log \frac{N_w}{N}).$$

We now compare the gain with mutual information below. Consider the training data as joint observation of (query, document) events, and assign the maximum likelihood estimates for the corresponding probabilities. Define a random variable X (Y) that takes the value 1 when the word w is in the query (document) and 0 otherwise. We have

$$P(XY = 11) \log \frac{P(XY = 11)}{P(X = 1)P(Y = 1)} = \frac{N_w}{N} \log \frac{N}{N_w}$$

and the above gain per document is nothing but

$$P(XY = 11) \log \frac{P(XY = 11)}{P(X = 1)P(Y = 1)} +$$

$$P(X = 1)P(Y = 1) - P(XY = 11).$$

Compare it with mutual information between X and Y:

$$\sum_{XY} P(XY) \log \frac{P(XY)}{P(X)P(Y)}.$$

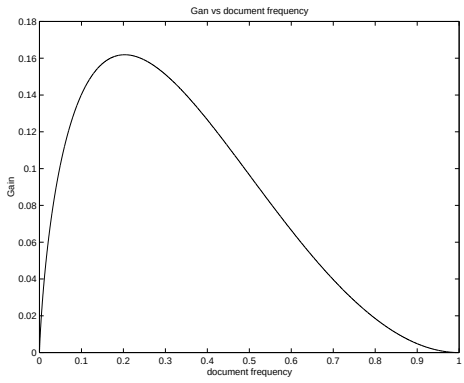
Both vanish when X and Y are independent. However, there is no notion of prior information (prior knowledge) in the framework of mutual information. But, the generalized Kullback-Leibler distance framework is built around a prior and the gain relative to this distance is comparable to mutual information when the prior is uniform in this setting.

Expressing the gain in terms of $f := N_w/N$, the proportion of documents containing the word, we get

$$\text{Gain} = f * (f - 1 - \log f).$$

We plot the gain as function of document frequency f and as a function of IDF.

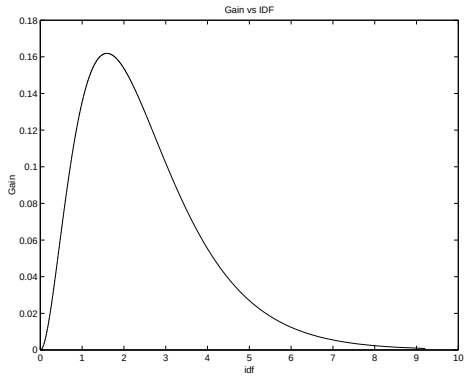
Figure 1: Gain vs Document Frequency



We see that the importance of a word is not linearly proportional to the IDF. Indeed, words that occur extremely rarely have very high IDF but very low gain as do words that occur in almost every document. This corroborates H.P. Luhn's observation that mid-frequency words have the highest "resolving power" (Salton and McGill, 1983), p. 62.

The above discussion deals with the weight of a unigram. How about the importance and optimal weight of a bigram? Information retrieval systems benefit by indexing on words as well as phrases (n-grams). But there are too many n-grams to index and therefore it is important to decide which n-grams are worth indexing. We limit our discussion to bigrams here. If we naively view a bigram

Figure 2: Gain vs IDF



as a unigram by gluing the words together, we get a high value for IDF of the bigram as $\log \frac{N}{N_{vw}}$. This assignment does not consider the fact the unigram IDF of v already discriminates documents that contain v. Intuitively, for the purpose of assigning bigram IDF for vw, we must redefine the universe of documents as the ones that contain v. Then, the bigram IDF is given as

$$IDF(vw) = \log \frac{N_v}{N_{vw}}$$

where N_v is the number of documents containing v and N_{vw} is the number of documents containing both v and w in consecutive positions (one can also define a corresponding position-independent feature). This is precisely the optimal λ we get for the feature ϕ_{vw} in the generalized Kullback-Leibler distance framework with a new prior

$$Q_0(c|x) = \frac{1}{N} e^{\lambda_v \phi_v(x,c)},$$

with $\lambda_v = \log \frac{N}{N_v}$. If the prior is as above, the value of adding the bigram to the model is then given by

$$\text{bigram gain} = N_{vw}(\frac{N_{vw}}{N_v} - 1 - \log \frac{N_{vw}}{N_v}).$$

For instance, "Humpty Dumpty" feature is useless once we have the "Humpty" feature in the prior.

We also get the same gain and weight as above when the prior has two non-overlapping features: a feature that examines the occurrence of the word v in both the query and the document, and a feature that examines the occurrence of w in the absence of v in both the query and the document.

Therefore, the above gain can be interpreted as the importance of the bigram when both v and w features are in the prior model. The gain should be negligible when the bigram vw does not improve retrieval likelihood over the component word features. We ranked bigrams in the Reuters-21578 corpus by their utility over the component word features.

We now briefly discuss the Reuters experiment. We processed documents with LEWISSPLIT="TRAIN" in the Reuters-21578 corpus, available at http://www.research.att.com/lewis. We processed 12869 documents containing 1.7 million words. The body of each document is lowercased and stripped of punctuation and then passed through a stop-word filter. Bigrams and unigram counts from the resulting document are accumulated. Since in a feature selection framework, we will not consider features with very low scores, we postulated only those bigrams when the component words have a generalized likelihood gain above a threshold. This is different from thresholding on the IDF as can be seen from the above plots. Out of the 473741 possibly bigrams, only 1272 passed the test. The postulated bigrams were ranked by the above gain. The top few and bottom few examples are shown in Figure 3. Please note that "per" was a stop-word. The top bigrams appear to be natural phrases in the domain whereas the bottom bigrams appear to be chance bigrams. The second column shows the generalized likelihood gain in milli-bits per document. The last column is the IDF of the bigram. In a retrieval system, we would index on all bigrams above a certain threshold of the gain.

The above computation of the weight for the bigram assumes that we do not retrain the weight of ϕ_v simultaneously when computing the weight of ϕ_{vw}. Following results are necessary to obtain a closed-form solution to the retraining of the weights of the unigram and bigram features relative to the uniform prior. First we note that a unigram and a bigram feature overlap. Overlapped features are more difficult to train than non-overlapping features. In the generalized Kullback-Leibler framework, the problem with non-overlapping features decouples into many single-parameter problems for which we displayed the closed-form solution above. The main idea in the next section is to transform the features so that they become non-overlapping and then to recover the solution to the original problem from that of the simpler transformed problem.

5 Non-overlapping binary features

We say that a set of binary features $\{\phi_k(x,c)\}$ is non-overlapping if $\sum_k \phi_k(x,c)$ is also binary-valued. We have the following two results which have no direct analog in the regular maximum entropy framework.

Proposition 2. The multi-parameter optimal solution to n non-overlapping binary features is simply obtained by solving n scalar binary feature problems independently, *all with the same prior* Q_0. That is, Opt $[\lambda_k]_1^n = [$ Opt $\lambda_k]_1^n$.

Figure 3: Utility of a bigram

Bigram	Gain (milli-bits)	IDF
year shr	59	2.23
pct year	53	2.72
pct last	42	3.18
cts share	39	1.63
pct interest	38	3.35
year net	37	3.21
billion dlr	37	1.89
pct billion	35	3.46
pct stock	35	3.49
year pct	34	3.36
year billion	33	3.42
stock exchange	33	1.84
last year	32	0.81
year company	31	3.53
net sales	30	2.52
u.s trade	29	2.94
pct company	29	3.78
cts net	28	0.84
u.s government	27	3.07
shares pct	23	2.40
due april	22	2.22
company stock	20	4.13
billion year	19	3.38
stock market	19	3.14
....	..	....
....	..	....
trade march	0.68	7.13
trade shares	0.68	7.13
trade year	0.68	7.13
interest government	0.68	7.13
interest shares	0.68	7.13
interest today	0.68	7.13
interest international	0.68	7.13

Proposition 3. For non-overlapping binary features, the multi-parameter optimal gain is the sum of the single parameter optimal gains.

However, the features ϕ_v and ϕ_{vw} are not non-overlapping. By transforming the features as below,

$$\begin{bmatrix} \psi_v \\ \psi_{vw} \end{bmatrix} := \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \phi_v \\ \phi_{vw} \end{bmatrix},$$

we get new features that are non-overlapping whose solution can be obtained in closed-form (assuming ψ_v is not identically zero) as

$$\begin{bmatrix} \lambda^{\psi}_v & \lambda^{\psi}_{vw} \end{bmatrix} = \begin{bmatrix} \log \frac{N}{N_{v\bar{w}}} & \log \frac{N}{N_{vw}} \end{bmatrix},$$

where $N_{v\bar{w}}$ is the number of documents that contain v but not w. It remains to recover the optimal weights for ϕ_v and ϕ_{vw} from the weights above.

We note that a linear-invertible transformation of the feature vector results in an equivalent optimization problem. Given a ϕ, let λ_* be the optimal solution with optimal value o_ϕ. We consider that $\lambda \in R^{1\times n}$. We denote the feature, its unique optimal solution, and the optimal value of the objective function by the triple $(\phi, \lambda_*, o_\phi)$.

Observation 3. Let A be an invertible matrix. Then, $(\phi, \lambda_*, o_\phi) \Leftrightarrow (A\phi, \lambda_* A, o_\phi)$.

From the above lemma, it follows that

$$\begin{bmatrix} \lambda^{\phi}_v & \lambda^{\phi}_{vw} \end{bmatrix} = \begin{bmatrix} \log \frac{N}{N_{v\bar{w}}} & \log \frac{N_{v\bar{w}}}{N_{vw}} \end{bmatrix}.$$

In the above derivations, it is assumed that there are only two features in the model. This is much like the assignment of IDF of a unigram that ignores co-occurrences of words. To take care of all correlations, one can solve the full multi-parameter problem with all unigram, bigram features using the improved iterative scaling algorithm. Indeed, term frequencies can also be taken into account in this framework, which is the subject of the next section.

6 Term Frequencies

This framework also allows for counting the number of occurrences of a given word in the document. We simply define an integer-valued feature for a word w as below:

$$\phi_w(x, c) = \begin{cases} \mathrm{tf}_{wc} & \text{if } w \text{ is in both } x \text{ and } c \\ 0 & \text{else} \end{cases}$$

where tf_{wc} is the number of times the word w appears in document c, that is, the term frequency of w in c. We assume there is one such feature for every word in the vocabulary. This results in the following score for a document given a query:

$$Q(c|x) = Q_0(c|x) e^{\lambda \cdot \phi(x,c)}$$

Taking logarithms, we have

$$\log Q(c|x) = \log Q_0(c|x) + \sum_{w \in x \cap c} \mathrm{tf}_{wc} \lambda_w.$$

Recall that λ_w turned out to be IDF of w when there is a single binary-valued feature in the model. Now we have many features and they are not binary-valued. Therefore the above scoring function is only analogous but not identical to the familiar TF-IDF scoring seen in the information retrieval literature. But, we can obtain the weights for term-frequency features in a principled manner by minimizing the generalized likelihood proposed in this paper. Another variation is to use document-length normalized term frequency in the definition of the feature, which will be real-valued.

7 Conclusion

The IDF of a word is optimal for document self-retrieval with respect to a generalized Kullback-Leibler distance. This conclusion is the result of viewing words independently and does not take into account that words co-occur. While IDF is cheap to compute, it must be modified when co-occurrence of words is taken into account. There is a general optimization framework where word co-occurrences can be taken into account and which produces the familiar IDF in the special case of a single feature. The framework is that of a relaxed maximum entropy or equivalently a relaxed log likelihood or Kullback-Leibler distance. The general framework opens up the possibility of assigning weights to more sophisticated lexical questions that is consistent with the popular notion of IDF.

Acknowledgments I thank R. T. Ward and J. S. McCarley for many valuable discussions. I also thank the anonymous reviewers for their comments.

References

Michael Collins, Robert E. Schapire, and Yoram Singer. 2000. Logistic regression, Adaboost, and Bregman distances. *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory.*

I. Csiszar. 1991. Why least squares and maximum entropy? an axiomatic approach to inference for linear inverse problems. *Annals of Statistics*, 19:2032–2066.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

John Lafferty, Stephen Della Pietra, and Vincent Della Pietra. 1997. Statistical learning algorithms based on Bregman distances. *Canadian Workshop on Information Theory*, pages 77–80.

R. Lau, R. Rosenfeld, and S. Roukos. 1993. Adaptive language modeling using the maximum entropy principle. *Proceedings of the ARPA Human Language Technology Workshop '93*, pages 108–113.

Kishore Papineni. 2000. A generalized Kullback-Leibler distance and its minimization. *IBM Research Report RC21815*, August. Also available at www.research.ibm.com/resources/paper_search.html.

S. E. Robertson and K. Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science*, pages 129–146, May-June.

G. Salton and M. J. McGill. 1983. *Introduction to Modern Information Retrieval.* McGraw-Hill.

K. Sparck Jones. 1973. Index term weighting. *Information Storage and Retrieval*, 9:619–633.

S. K. M. Wong and Y. Y. Yao. 1992. An information-theoretic measure of term specificity. *Journal of the American Society for Information Science*, 43:54–61.