13

Skipping-Based Collaborative Recommendations inspired from Statistical Language Modeling

Geoffray Bonnin, Armelle Brun and Anne Boyer
LORIA – Nancy Université
France

1. Introduction

Due to the almost unlimited resource space on the web, efficient search engines and recommender systems have become a key element for users to find resources corresponding to their needs. Recommender systems aim at helping users in this task by providing them some pertinent resources according to their context and their profiles, by applying various techniques such as statistical and knowledge discovery algorithms.
Recommender systems are usually classified into content-based recommendation and collaborative filtering. Content-based recommendations (Balabanović & Shoham, 1997; Zhang et al., 2002) are performed by identifying resources similar to the ones a user appreciated, based on their content. One of the main limitations of these systems is that the efficiency is highly dependent on the domain. Indeed, it is very efficient for textual resources but not for resources such as pictures, videos, etc. Another limitation is that only resources similar to already rated resources can be recommended. Collaborative Filtering (Das et al., 2007; Goldberg et al., 1992), consists in recommending to users resources other users with similar tastes liked in the past. The content of the resources does not need to be considered, and the aforementioned limitations are not present. However, collaborative filtering has its own limitations, the most important being data sparsity and cold start (Park et al., 2006; Schein et al., 2002). Most of recommender systems only use ratings to predict if a user will appreciate some resource, and to provide recommendation lists by selecting the highest ratings predicted, or the most similar resources to resources a user already rated (Adomavicius & Tuzhilin, 2005). The quality of the recommendations may thus be enhanced by using other criteria.
Such a criterion is the context, which can be geographical, meteorological, social, cultural, etc. For instance, a user may like to eat his favorite dish at home but not in a restaurant, or at lunch but not at breakfast. The importance of using context for recommendations have been studied on a movie rating dataset in (Adomavicius et al., 2005). Among with the ratings, users were asked when, where and with whom the movie was seen. Results showed that using a combined form of a reduction-based collaborative filtering method to include contextual information in the model, the accuracy could be significantly outperformed compared to a standard memory-based collaborative filtering algorithm.
The order in which users consult or consume resources, which is referred to as sequences of consultations, is such a contextual criterion. For instance, one usually must have seen the first episodes of a television series to appreciate the last ones. In this chapter we focus on

this particular form of context. The question is thus: how to take advantage of sequences to recommend the best possible resource?
The appropriateness of considering sequences is domain dependent: for instance, it seems of little help in domains such as on-line movie stores, in which user transactions are barely sequential; however it is especially appropriate for domains such as web navigation, which has a sequential structure. This was shown in (Zimdars et al., 2001), in which several techniques are used to transform sequences into a representation that can be used by traditional collaborative filtering algorithms. This representation makes the resulting model almost equivalent to a Markov model. A decision tree model has been used to perform tests on a browsing dataset. Results show a clear enhancement of the results using a sequential configuration instead of a classical collaborative filtering configuration.
Predicting future surfing paths is useful for many purposes such as web page research (Tan & Kumar, 2002), web page recommendations (Nakagawa & Mobasher, 2003), latency reduction (Schechter et al., 1998) or arrangement of the links among a website (Chi et al., 1998). That is why it has been widely studied. Such studies do not necessarily include ratings, for instance sequential patterns (Nakagawa & Mobasher, 2003) or Markov models (Borges & Levene, 2005; Deshpande & Karypis, 2004; Eirinaki & Vazirgiannis, 2007; Pitkow & Pirolli, 1999), although some other do (Trousse, 2000).
Web predictive modeling usually attempts to provide a tradeoff between accuracy, space and time complexity, and coverage (Deshpande & Karypis, 2004; Pitkow & Pirolli, 1999). However, few of these models possess features able to provide robustness to noise. Noise can occur when users do navigation mistakes, parallel navigations, open pages in new tabs, return to previous pages, etc. The amount of noise may vary depending on the domain. For instance, a website designed by an experimented webmaster usually induces less navigation mistakes than personal web pages within a web hosting service.
A study of statistical language modeling allowed us to notice that several similarities exist between web navigation and natural language (Boyer & Brun, 2007). Many statistical language models have been studied in the past decades with success, and most of them take into account the order of the words. We thus propose to draw inspiration from these models to compute recommendations.
We propose a new model inspired from the n-gram skipping model of statistical language modeling (Goodman, 2001) to compute recommendations in the frame of web navigation. This model exhibits several advantages: (1) It is robust to noise, (2) It has both a low time and a low space complexity while providing a full coverage, (3) Weighting schemes are used to alleviate the importance of distant resources, (4) A significant improvement of accuracy compared to state of the art models is provided.
In the first section, we will address the general issue of applying statistical language modeling to web navigation. The second section presents our Skipping-Based Recommender or SBR model. Tractability is then discussed in the third section, and robustness to noise in the fourth section. Last, we conclude the chapter.

2. Modeling Web Navigation as a Natural Language

In this section, we provide a detailed study of web navigation and natural language to explicit their similarities. We first provide an overview of web predictive modeling and natural language modeling. We then show similarities and differences between both domains, and present a discussion about which statistical language models seem to be the most appropriate

for web navigation, and which adaptations seem necessary to maximize their efficiency for this domain.

2.1 Web Predictive Modeling

Recommending resources to users in the frame of web navigation is one of the most important tasks of web usage mining. Web usage mining can be defined as "the process of applying data mining techniques to the discovery of usage patterns from web data" (Srivastava et al., 2000). In this domain, recommending resources to users is referred to as predictive user modeling (Nakagawa & Mobasher, 2003) or predictive modeling (Pitkow & Pirolli, 1999).
The data processed usually consists in two sets: a set of distinct resources $R = \{r_1, ..., r_{|R|}\}$ and a set of sessions $S = \{s_1, ..., s_{|S|}\}$ where each s_i is a sequence of resources from R, *i.e.* $s_i = (\rho_1^i, ..., \rho_{|s_i|}^i)$ with $\rho_j^i \in R$.
Two approaches are predominant: sequential patterns and Markov models. We thus present them in this section.

2.1.1 Sequential Patterns

One way of exploiting the order of past actions to predict future ones, is the use of sequential patterns (Agrawal & Srikant, 1995; Lu et al., 2005), which is the sequential form of association rules. Association Rules have been initially used for mining supermarket basket (Agrawal et al., 1993) to extract information about purchased items dependencies. An association rule is made up of items commonly purchased together in a transaction, where a transaction is a set of items.
An association rule is an expression of the form $X \rightarrow Y$, where X and Y are sets of items. X is called the antecedent and Y the consequent. An association rule means that, in one transaction, when users have purchased all resources in X then there is a high probability that they will purchase Y. Using association rules in the frame of web usage modeling thus enables to take into account non-ordered sets of resources in the history. Sequential patterns are more constrained than association rules due to the order taken into account. They thus represent more accurate information about user behavior. The sequences considered can be ordered lists of sets of resources (*e.g.* $(\{a, b\}, \{c, d, e\})$). However in this chapter, we only focus on sequences of single resources (*e.g.* (a, b, c, d, e)).
Usually, the consequents considered in sequential patterns have a size of 1 (Nakagawa & Mobasher, 2003). So, a sequential pattern can be denoted by $X \circ Y$, where $X \circ Y$ is the concatenation of X and Y, X is a sequential antecedent of any size, and Y a consequent of size 1.
Both models are first built by browsing a training corpus and counting the sets of resources or sequences of resources. Then, during the recommendation step, all possible antecedents in current user's navigation history are compared to the antecedents in the model. If some antecedents match, then the corresponding consequents are recommended. However, several antecedents of different sizes may match the history, which raises the question of combining the rules. A score can be assigned to each consequent according to each corresponding matching antecedent. This can be done in several ways:

- Maximum confidence policy: only the rule having the highest confidence is used (Sarwar et al., 2000; Wang et al., 2005);
- Sum policy: the sum of the confidences is computed and associated to the corresponding consequents. Then, the consequent having the highest value is recommended (Kim & Kim, 2003);

- Maximum length policy: the rules having the longer antecedent are used to provide the recommendations (Nakagawa & Mobasher, 2003). This scheme is analogous to the all-k^{th}-order Markov model (*cf.* next section). It can be combined with maximum confidence policy or sum policy.

There are two types of sequential patterns: closed and open (Mobasher, 2007). Closed sequential patterns (Jianyong et al., 2007) identify contiguous sequences, while open sequential patterns (Ayres et al., 2002) identify non contiguous sequences. Looking for open sequences of unlimited sizes induces a huge amount of combinations. So the step of pattern discovery has to limit the size of the patterns to discover. Thus a sliding window with a fixed size is usually used during the pattern discovery step as well as during the recommendation step. However, the time complexity induced is still high. As they induce less combinations, the time complexity of closed sequential patterns is lower, but still high.
Space complexity can be reduced by integrating only the rules with a high *support* and *confidence* in the model. This was already the case for association rules, and was coped using the Apriori algorithm, which is an incremental algorithm (Agrawal et al., 1993). This algorithm has first been adapted to sequential patterns by (Srikant & Agrawal, 1996) and is referred to the Generalized Sequential Pattern algorithm or GSP algorithm. It is based on incremental pruning of low support and confidence patterns. Given a set of sessions $S = \{s_1, ..., s_{|S|}\}$, the support of a pattern $X \circ Y$ is defined as:

$$supp(X \circ Y) = |\{\sigma \in S | X \circ Y \subseteq \sigma\}|$$

where each σ is a subsequence of size D in S. The confidence of the sequential pattern $X \circ Y$ is defined as:

$$conf(X \circ Y) = \frac{supp(X \circ Y)}{supp(X)}$$

The algorithm first counts all sequences of size 1, and prunes the less frequent ones. It then builds sequences of size 2 using the remaining sequences of size 1, computes the corresponding counts and prunes the less frequent sequences. The algorithm continues until the sequences reach some maximum length. The supports thresholds used are usually the same whatever is the length of the considered sequence.
Selecting high confidence and support rules induces a lower space complexity and a higher accuracy; however, it induces a lower coverage of longer patterns too. Indeed, although a recommendation can always be provided using sequential patterns of size 1 (antecedent of size 0 and consequent of size 1), selecting few rules induces that longer antecedent match more rarely the previous user actions (Nakagawa & Mobasher, 2003).
In (Nakagawa & Mobasher, 2003), an empirical study comparing association rules, closed and open sequential patterns is provided. Results show that association rules and open sequential patterns are more suitable for short sessions and sites with a high degree of connectivity, while closed sequential patterns are more suitable for longer sessions. However the experiments have been done using small window sizes (3 and 4), and it is possible that higher window sizes lead to a different conclusion.

2.1.2 Markov models

Markov chains (Rabiner, 1989) model relationships between resources based on an independence assumption between past states and the present state. In the frame of web navigation they are used to predict the next resource according to the present state (the k previously

browsed resources), which is referred to as Markov models of order k or k^{th}-order Markov models. Although simple, Markov models provide surprisingly accurate recommendations.
Markov models are built the same way sequential patterns are, *i.e.* by browsing a training corpus and counting sequences of resources of size $k+1$. The recommendation step is similar to the one of sequential patterns too: the previous actions are compared to the states in the model, and if some state matches, then the corresponding resource is recommended.
The use of Markov models usually involves a tradeoff between accuracy and coverage (Pitkow & Pirolli, 1999). Coverage is the percentage of cases where a state matching current history can be found in the model to recommend a resource. By pruning the less frequent elements, a better precision can usually be reached; however, the more elements are pruned, the less matching histories can be found during the recommendation step, which results in a lower coverage. Notice that contrary to sequential patterns, the number of possible states is low enough to perform a straightforward pruning, after having performed the training step.
Another way to enhance the accuracy is to increase the value of k. Indeed, a state having a higher length contains more information about user's past actions. However, above some value it becomes difficult to find a large enough training data to build the model. If the training dataset is too small, the resulting model will cover fewer cases and may even provide a lower accuracy; if a large enough training data can be found, the model may have a too high space complexity. That is why the length of the states is usually low.
One way to provide both accuracy and coverage is to use various Markov models having various orders. For example, one can try to provide a recommendation using a Markov model of order 3, and if no matching history can be found, try a Markov model of order 2, and so on, until a recommendation can be provided. In the worst case, a Markov model of order 0 is used, which corresponds to the overall probability of one single resource, without considering previous resources. Using such a scheme, a full coverage can be reached, while providing a good accuracy in the recommendations. This scheme is called the all-k^{th}-order Markov model (Pitkow & Pirolli, 1999), and is one of the best performing predictive models of the state-of-the-art. Notice that under the same pruning conditions, it is similar to closed sequential patterns.
Several studies have been done to cope with space complexity. In (Deshpande & Karypis, 2004), three pruning schemes are used to alleviate the state complexity: a support pruning scheme in which the same threshold is used for all of the Markov models, a confidence pruning scheme in which states are discarded if the difference of probability between the two most prominent resources is not statistically significant and an error pruning scheme using a validation dataset. (Borges & Levene, 2005) propose to transform first-order Markov models into a single model representing Markov models of variable orders by using cloning operations. This lowers time and space complexity while providing a full coverage and a good accuracy.
Instead of trying to deal with tradeoffs between accuracy, space and time complexity and coverage, some studies simply combine Markov models with some standard recommendation models to enhance the precision of the recommendations. In (Trousse, 2000), a case-based model is used to predict users' navigation behavior. The main feature of the model is the inclusion of past sequences in the cases, which is referred to as time-extended situations. Two sequential features are used to represent the cases. The first one corresponds to the last three browsed pages. The second is the sequence of the past pages having a high implicit rating. The main drawback of such a model is the coverage. Indeed, using the last three browsed pages is similar to the present states of a Markov model of order 3. As said previously, in such a case the number of possible states is generally high, and usually results in a low coverage. Besides,

a maximum coverage of 50% is reached in their experimentations. In (Eirinaki & Vazirgiannis, 2007), a PageRank-based model that includes a usage based personalization vector has been experimented. The personalization vector used is similar to a Markov model of order 1 whose transition values are computed according to websites' actual structures. Two variants of the model are put forward. The first one is called *l*-UPR (localized Usage-based PageRank) and consists in using current user's sessions to compute the personalization vector. The second is called *h*-UPR (hybrid Usage-based PageRank) and is a combination of the UPR model with a standard Markov model. In such a configuration, recommendations are based both on current users' usage data and actual website structure.

2.2 Statistical Language Modeling

The issue of Language Modeling is to compute the probability of a word w_i given its history $h = (w_1, ..., w_{i-1})$. The data processed usually consists in two sets: a set of distinct words called the vocabulary $V = \{v_1, ..., v_{|V|}\}$ and a set of sentences $S = \{s_1, ..., s_{|S|}\}$ where $s_i = (w^i_1, ..., w^i_{|s_i|})$ with $w^i_j \in V$. For complexity and feasibility reasons, the vocabulary is usually previously fixed.

As too long histories are computationally intractable, all existing statistical language modeling techniques assume some form of independence among different portions of the data. This results in approximated probabilities which can be calculated statistically using a training data (Rosenfeld, 2000). Surprisingly, statistical techniques have been shown to definitely perform better than linguistic rule-based techniques (Banko & Brill, 2001; Fleischman et al., 2003; Och & Ney, 2001).

The two predominant statistical models are the *n*-gram model and the trigger model, which are presented in the following.

2.2.1 *n*-gram model with skipping

Markov models are also used in the domain of statistical language modeling, in which they are referred to as *n*-gram models (an *n*-gram model is similar to a Markov model of order $n-1$). *n*-grams even represent the cornerstone of statistical language modeling (Rosenfeld, 2000).

As for web usage mining, in practice, $n = 3$ or 4, rarely 5. As well, the coverage problem is present too. An experiment performed in the 1970s by IBM puts forward this phenomenon. In this experiment, a text containing one thousand distinct words (the vocabulary) was divided into a training set of $1,500,000$ words and a test set of $300,000$ words. Then a trigram model built on the training set only covered 77% of the test set.

Many techniques have been used to enhance their efficiency, among which smoothing, clustering, mixture, etc. (Goodman, 2001). One of these improvements is skipping and is based on the fact that the larger the *n*-grams, the less matching histories can be found (due to the size of the training dataset). Skipping simply consists in not considering a resource: the resource is skipped. For example, given the sequence (a, b, x, y, z, c, d) and $n = 3$, instead of considering only contiguous raw triplets as (a, b, x) or (y, z, c) (as standard *n*-gram models), skipping allows to also consider triplets as (a, x, d), (a, b, c) or (b, c, d).

There are two ways of using skipping: by interpolating submodels (Goodman, 2001), and by merging the counts.

- Interpolation of submodels

The first way of using skipping consists in interpolating skipping submodels. For instance, the probability of a word w_i given the history h can be given by the following equation:

$$P(w_i|h) = \alpha P(w_i|w_{i-2}, w_{i-1}) + \beta P(w_i|w_{w-3}, w_{i-1}) + \gamma P(w_i|w_{i-3}, w_{i-2}) \quad (1)$$

where $0 \leqslant \alpha \leqslant 1$, $0 \leqslant \beta \leqslant 1$, $0 \leqslant \gamma \leqslant 1$ and $\alpha + \beta + \gamma = 1$. Using such a scheme thus allows to handle an over probability of a 4-gram-like model using trigram models, and to find more matching histories than would find a raw 4-gram model. The configuration corresponding to $P(w_i|w_{i-3}, w_{i-2})$ is also known as a distance-2 trigram, *i.e.* a long-distance n-grams with $n = 2$. Such models were studied in (Huang et al., 1993) and did not provide significant improvements. In (Goodman, 2001) however, it has been shown that skipping of trigrams (skipping involving 3 elements) represents a good technique to use if the training data is small. The interpolation of submodels has the advantage of allowing an accurate weighting of the respective skipping configurations. Indeed, as separate submodels are built, it is possible to estimate the importance and usefulness of each skipping configuration on the recommendation process. The major drawback of interpolating submodels is that it implies a larger number of n-grams. This last problem becomes worse when considering an even larger history: the larger the history, the larger the number of submodels.

- Merging the counts

The second way of using skipping is to merge all skipped n-grams occurrences so that they are all stored in the same list. This has the advantage of lowering the space complexity. For instance, given the training sequence (a, x, b, c, a, b, y, c), it is possible to detect three occurrences of the trigram (a, b, c). Such counts can then be stored in the same trigram counts in one single list. However, once stored in the list, it is impossible to determine if a n-gram has often been encountered in a contiguous configuration, or if it was almost always encountered in another particular skipping configuration. The interpolation of mixed submodels thus represents a more accurate modeling.

Another advantage of merging counts is that it allows some skipped n-grams of the model to be used by other skipping configurations. For example, given a trigram (a, b, c). It is possible that this trigram is found several times in some parts of the training data together with some noise resources x_i between a and b: (a, x_i, b, c). Then, when making recommendations, it is possible that a and b are found in the history before some other noise resource y: (a, b, y). In that case, during the training step, skipping is performed on the second resource (x_i) whereas during the recommendation step it is performed on the third (y), and the resource c can be recommended, which is not possible using the interpolation presented above. This feature is interesting because it induces a better coverage; however it represents a less accurate modeling of the data.

2.2.2 Trigger model

One of the first introduction of trigger models is (Rosenfeld, 1994). It was designed based on the observation that some information exists beyond the usual scope of contiguous n-grams. Trigger models are made up of highly correlated pairs of words, the first one being the trigger, and the second one the triggered word.

The selection of the trigger pairs is usually performed using the mutual information that measures the quantity of information provided by a trigger word A to a triggered word B. It is usually evaluated as follows:

$$MI(A,B) = p(A,B)\log\frac{p(B|A)}{p(B)} + p(A,\overline{B})\log\frac{p(\overline{B}|A)}{p(\overline{B})} + p(\overline{A},B)\log\frac{p(B|\overline{A})}{p(B)} + p(\overline{A,B})\log\frac{p(\overline{B}|\overline{A})}{p(\overline{B})} \quad (2)$$

where A denotes the presence of A, $\overline{A}$ denotes the absence of A, $P(A)$ the probability of A, $p(A,B)$ the probability that A and B are found together and $p(B|A)$ the conditional probability of B given A. This selection is usually performed using a fixed size sliding window.
Once the triggers have been selected, they are used to refine n-gram models (Chen & Chan, 2003; Rosenfeld & Huang, 1992). However, several triggers may match the history and have to be combined. This can be done in several ways that provide similar results : choosing only the trigger pair having the highest mutual information, adding the mutual information values of the trigger pairs, etc. (Rosenfeld & Huang, 1992).
As well, the way triggers are integrated in the n-gram model has to be determined. This can be done by using an interpolation, by enhancing the probabilities of the corresponding n-gram when a word is triggered, etc. (Rosenfeld & Huang, 1992).
Such models usually take into account only relationships between two words, although they can be applied to longer triggers, which could lead to more accurate models (for the same reason a higher order Markov model is more accurate). In (Chen & Chan, 2003) a model called the multi-word trigger model is studied. In this model, the triggering elements consist in pairs of words and here again, only the most correlated triplets of words are integrated using the mutual information. However, this did not lead to significant improvements.

2.3 Web navigation and Natural Language Similarities

In several points, web navigation predictive modeling is similar to statistical language modeling. This is particularly obvious when focusing on the respective corpus features.

- Words can be considered as being similar to resources (ignoring the content, just considering them as identifiers);
- Statistical language models use a vocabulary made up of words which can be viewed as being similar to the set of distinct resources of the web or a website.
- A sentence can be considered as being similar to a session;
- The presence of a word in a sentence depends on its previous words, as well as the consultation of a resource depends on the preceding resource consultations.
- Both domains provide large datasets that can be used to train statistical models.
- As can be noticed in the previous sections, both domains have been efficiently modeled using n-grams (n-gram models are equivalent to Markov models of order $n-1$). Thus both domains seem to allow a similar independence assumption.

Given these similarities, we can naturally think of exploiting statistical language modeling techniques for web recommendation. Statistical language modeling was studied far previously to web recommendation, and a lot of efficient models have been studied, it thus provides interesting perspectives.
However, two main differences exist between natural language and web navigation: (1) it is possible to have several web navigations overlapped, which would correspond to mixed sentences in natural language which does not exist, (2) natural language is governed by strong

constraints: each word and its localization in a sentence is important; navigation is less constrained and should be processed with more permissive models.
Thus statistical language modeling cannot be applied directly for web recommendation. Parallel navigations have to be handled. As the resources of one session can be relatively distant, the history considered should be longer than those of classical language modeling.
The second difference is problematic too. A more constrained corpus means that a light model can be build from it, and thus a less constrained data means that the resulting model is heavier. As the web contains a large number of resources, a rather light model would be welcome.
Hence, exploiting statistical language models induces an adjustment of the algorithms in order to provide a light and permissive model.

2.4 Exploiting statistical language modeling for web recommendation

In this section we discuss the exploitation of the aforementioned statistical language models for web usage mining. The goal is to find a model providing a high accuracy in the recommendations, a high coverage and a good robustness to noise while being tractable, which cannot be provided by classical statistical language modeling.
As previously said, using closed sequences (*e.g.* Markov models) makes it impossible to handle noise. One solution is to use open sequential patterns, but then the number of possible patterns is very high which leads to a high complexity.
A first possibility is the use of trigger models. Trigger models allow to consider distant elements, and only the most informative pairs are included in the model, which allows to discard noise. Indeed, if an element corresponds to noise, the impact of all other elements within the window will compensate its impact. The use of mutual information provides another interesting feature. Indeed, when using conditional probabilities, the most frequent resources are more likely to be recommended, although such resources may not be of major utility for a user. For instance, the home page of a website is usually the most visited one, but may not be the most interesting page to recommend to a user. Using the mutual information measure has a different effect of using conditional probabilities: the most frequent words are less likely to have a large mutual information value. However, as well as for natural language, they cannot be used alone. Indeed, a rare resource having a high correlation to a previous resource may be recommended, which may not be useful for a user.
Trigger models should thus also be combined with n-grams, as classically used in statistical language modeling in order to take advantage of both models. Such a configuration has been tested by (Pavlov et al., 2004). The models presented consist in mixtures of sub-models. In particular a bigram model (n-grams with $n = 2$) is combined with a trigger model. Both sub-models are interpolated using coefficients computed according the Expectation Maximization Algorithm (Dempster et al., 1977) on a validation set. Depending on the considered data, this algorithm may take too long to converge to an optimal solution. (Pavlov et al., 2004) thus propose to use a fast clustering algorithm based on users' navigation sequences. Using such a framework allows to take into account distant resources and to provide a high coverage while having a low time and space complexity. However, the use of a bigram model provides less accurate recommendations, and if the previous resource in the history is noise, then a bad recommendation is likely to be provided. This may not be compensated by the combination with a trigger model, as the mutual information may not be appropriate used alone, and should only be used as a complement.
An alternative is to use n-gram models with skipping. It allows distant resources to be taken into account, while using conditional probabilities. It is very close to open sequential pat-

terns; the main difference is that it is usually performed with a fixed value of n, and has a lower space and time complexity. The resulting model is thus tractable and robust to noise. Coverage depends on the considered data. With a fixed value of n, it is obvious that an n-gram model with skipping provides a better coverage than a raw n-gram model. Depending on the considered data and the value of n, it is possible that the coverage is not full, which can only be determined experimentally. An n-gram model with skipping was used in (Shani et al., 2005) to initialize a Markov Decision Process recommender system. When building the model, in addition to the raw n-gram counts, weighted occurrences of skipped n-grams are added to the counts. The skipping is performed only between the next to last and the last resource of the n-grams, and the occurrences are weighted according to an exponential decay scheme. This n-gram model has been compared to a dependency network based model in which the local distributions are probabilistic decision trees. Although these algorithms are among the most competitive, the skipping-based model reached better results. However, the skipping is applied only during the training step, which has been shown to provide less accurate recommendations (Bonnin et al., 2008).
A last possibility is to combine a trigger model with an n-gram model with skipping. To the best of our knowledge, such a configuration has never been studied in the frame of natural language. This is because the strong constraints of natural language make raw n-gram models very efficient, and they just need to be refined using distant information. However, as argued above, web navigation is far less constrained and the combination of both models provides an interesting alternative. Indeed, the complementarity of mutual information and conditional probabilities may even enhance the accuracy of the recommendations. In the following of this chapter we focus on the previous configuration (n-grams with skipping).

3. The Skipping-Based Recommender

As shown in the previous sections, predicting user behavior involves tradeoffs between complexity, predictive accuracy and coverage. Sequential patterns handle distance between the resources, but induce a huge number of sequences. The all-k^{th}-order Markov models, as to them, induce fewer sequences, lead to a high coverage, but still need a high storage space and do not allow distance between resources, which does not allow robustness to noise. In statistical language modeling the use of skipping in n-gram models is a way to benefit from the accuracy of n-gram models while handling distant resources as trigger models do, which leads to a high coverage and a low time and space complexity. Due to these advantages, the recommendation algorithm we propose is an n-gram model with skipping and is called the Skipping-Based Recommender or SBR.
When using skipping, the elements that can be skipped have to be determined and the size of the skipping has to be fixed. We thus present several possible skipping variants. We then present the weighting schemes we apply in order to alleviate the importance of distant resources. Last we describe the recommendation process of the SBR model.

3.1 Skipping Variants

In the skipping variants we study in this article, we consider that when an element can be skipped, the size of the skipping is limited to the size of the window used (similar to the sliding window used in association rules and Markov models).

3.1.1 Shani's skipping

The first skipping variant we study is the one used in (Shani et al., 2005). It consists in allowing skipping only for the last element of the n-grams, all other elements being contiguous.
For example, let $n = 3$, and the navigation sequence: (a, b, x, y, z, c, d) where (a, b, c, d) and (x, y, z) correspond to overlapping navigations. This variant allows to consider triplets as (a, b, y) or (a, b, c) and also raw triplets as (a, b, x) and (z, c, d). It is thus able to capture distant elements of a sequence if the last element corresponds to the continuance of a previously initiated navigation as for the triplet (a, b, c). The elements between (a, b) and c are here considered as elements of another navigation, but may also be considered as noise.
However, this skipping variant is not able to capture a navigation overlap if the two last elements correspond to the continuance of a previously begun navigation (a step after the previous configuration): for example, the triplet (b, c, d) cannot be handled as b and c are not contiguous.

3.1.2 Full skipping

The full skipping variant goes a step further by allowing skipping between all the elements of the n-gram. It makes the resulting model almost equivalent to sequential patterns, for instance the one proposed in (Nakagawa & Mobasher, 2003). The main difference is that the SBR model considers only sequences of size n while sequential patterns usually handle variable size patterns. This variant has several advantages. First, a high amount of n-grams is processed, which provides a better coverage. Second, this skipping captures parallel navigations, noise and approximate sequences, wherever these unexpected actions are, and whatever is their size. However, it can be viewed as a too permissive variant and the size of the model rapidly grows.

3.1.3 Enhanced skipping

We designed this new variant especially to take into account noise and parallel navigations without inducing a high complexity. The first variant can handle only noise in the last element. The second variant handles noise everywhere in the navigation, which may be too permissive. We propose here a variant that can be considered to be between both. It allows the consideration of two configurations simultaneously: skipping the last or the first element of the n-gram, which enables noise either in the first part of the n-gram, or in the last part, but not both.
For instance, given the previous example, it becomes possible to handle both cases (a, b, c) and (b, c, d) but not (a, x, c).

3.2 Weighting Schemes

We argue that skipped n-grams handled by the aforementioned skipping variants cannot be considered in the same way than raw n-grams (contiguous n-grams), and thus propose to weight them.
We present in this section several weighting schemes that can be used to take into account these skipped n-grams. Let d_i be the distance between the i^{th} element and the last element of the trigram, and D the size of the window.
In order to show the benefits of weighting, we first propose to not use any weighting, as done for sequential patterns (Nakagawa & Mobasher, 2003). This weighting scheme is also similar to most of the trigger-based models: whatever is the distance between the elements of the n-

grams, they all have the same weight. In this case, the weighting scheme is referred to as the **No Weighting** scheme. The weight $w(d_1, ..., d_{n-1})$ of a given skipped n-gram is defined as:

$$w(d_1, ..., d_{n-1}) = \begin{cases} 1 & \text{if } d_1 \leq D \\ 0 & \text{else} \end{cases} \quad (3)$$

However, the recommendation impact of a skipped trigram should be lower than the one of a raw n-gram, due to the distance. We consider that the more a resource is distant, the more its influence is low, and the less the corresponding skipped n-gram is influencing. Thus, we propose to apply to a skipped n-gram a weight inversely proportional to the distance D. The following weight decreases linearly according to the distance. In this case, the weighting scheme is referred to as the **Linear Decay** weighting scheme. The weight becomes then:

$$w(d_1, ..., d_{n-1}) = \begin{cases} -\frac{d_1}{D} + 1 & \text{if } d_1 \leq D \\ 0 & \text{else} \end{cases} \quad (4)$$

Another way to perform this decrease is to decay exponentially the weight as proposed by (Shani et al., 2005). Using such a weighting scheme makes the value decrease faster. In this case, the weighting scheme is referred to as the **Single Exponential Decay** weighting scheme, and is defined as follows:

$$w(d_1, ..., d_{n-1}) = \begin{cases} 2^{-d_1} & \text{if } d_1 \leq D \\ 0 & \text{else} \end{cases} \quad (5)$$

This last scheme is sufficient for Shani's skipping variant. Indeed, only the last resource can be skipped, and it is not necessary to consider all the distances between the resources. In the enhanced and the full skipping variant however, other distances between the elements of a skipped n-gram may vary, and should be considered to compute the weightings. We thus propose to apply to skipped n-grams a weight that depends on the distance between each element of the n-grams and the resource to predict.
For example, applied to the sequence (a, b, x, y, z, c, d) with $n = 3$, triplets (a, b, d) and (a, c, d) should not have the same weight, even if the first element of both triplets is equidistant from the last element. Moreover the weight of (a, c, d) should be higher than the weight of (a, b, d) as the intermediate resource c is closer to d than b is. In this case, the weighting scheme is referred to as the **Multiple Exponential Decay** weighting scheme. The weight we propose to use is the following:

$$w(d_1, ..., d_{n-1}) = \begin{cases} \frac{1}{n-1} \sum_{i=1}^{n-1} 2^{-d_i} & \text{if } d_1 \leq D \\ 0 & \text{else} \end{cases} \quad (6)$$

Given the previous skipping variants and weighting schemes, the processing of recommendations can be done. The SBR model relies on the following steps:

Step 1: Training the model on a corpus to determine the counts of the n-grams

Step 2: Computing the probabilities of the n-grams based on their counts

Step 3: Running the model to recommend the pertinent resources to the active user given his/her stream of navigation

3.2.1 Training

In the training phase, raw n-grams and skipped n-grams are trained on the input data. The question now is how to mix the skipped n-grams and the raw n-grams. We propose to simply add the occurrences of skipped n-grams (weighted by the weighting schemes of the previous section) to the occurrences of raw n-grams (contiguous n-grams) as in (Chan & Goodman, 1998).

The weighted occurrences of the skipped n-grams are added to the counts of their corresponding raw n-grams. Algorithm 1 presents how these counts are computed when using the full skipping variant and $n = 3$. The count of the skipped trigram is denoted by $C(\rho_i, \rho_j, \rho_k)$.

Data: a set S of navigation sessions
Result: a list of trigrams associated with their occurrences
$trigramlist \leftarrow ()$;
for each session $s = (\rho_1 \dots \rho_{|s|})$ in S **do**
  for $i \leftarrow 1$ to $|s| - 2$ **do**
    for $j \leftarrow i + 1$ to $\min(i + D, |s| - 1)$ **do**
      for $k \leftarrow j + 1$ to $\min(j + 1 + D, |s|)$ **do**
        $trigram \leftarrow (\rho_i, \rho_j, \rho_k)$;
        $d_1 \leftarrow k - i - 1$;
        $d_2 \leftarrow k - j - 1$;
        if $trigram$ is in $trigramlist$ **then**
          $C(\rho_i, \rho_j, \rho_k) \leftarrow C(\rho_i, \rho_j, \rho_k) + w(d_1, d_2)$;
        else
          $C(\rho_i, \rho_j, \rho_k) \leftarrow w(d_1, d_2)$;
        end
      end
    end
  end
end

Algorithm 1: Computing counts of trigrams with skipping using the full skipping variant

3.2.2 Computing the probabilities of the n-grams

Given the n-grams counts from the training phase, the conditional probabilities have to be computed. Let the n-gram $(\rho_{i-n+1}, ..., \rho_i)$. The probability of the resource ρ_i given $(\rho_{i-n+1}, ..., \rho_{i-1})$ is computed as follows:

$$P(\rho_i \mid \rho_{i-n+1}, ..., \rho_{i-1}) = \frac{C(\rho_{i-n+1}, ..., \rho_i)}{C(\rho_{i-n+1}, ..., \rho_{i-1})} \tag{7}$$

where $C(\rho_{i-n+1}, ..., \rho_i)$ is the count of the skipped n-gram $(\rho_{i-n+1}, ..., \rho_i)$.

3.2.3 Recommending

The recommendation step consists in predicting the next resource r_i given the $D - 1$ previous resources in the session $(\rho_{i-D+1}, ..., \rho_{i-1})$. For each resource in the set of distinct resources of the data $R = \{r_1, ..., r_{|R|}\}$, a score is computed according to each possible skipping state σ.

This score is a weighted form of the probability that at least one of the skipping states leads to resource r_j. The score is given by the following formula:

$$q(r_j, h) = 1 - \prod_{\sigma} \Big(1 - P(r_j \mid \sigma) \cdot w(d_1, ..., d_{n-1})\Big) \tag{8}$$

where $P(r_j|\sigma)$ is the probability of r_j given the skipping state σ, $w(d_1, ..., d_{n-1})$ the weighting of the skipping n-gram $(\sigma \circ r_j)$ according to the distances $d_1, ..., d_{n-1}$ between its elements. The skipping states σ considered depend on the skipping variant chosen.
For instance, if a user has browsed the following resources:

388 401 55 359 325 369 381 368 366 60 72

Then if the window size is $D = 10$, the resources considered are the following: 55, 359, 325, 369, 381, 368, 366, 60 and 72 (the 9 previous resources). If the skipping variant is the enhanced skipping and $n = 3$, than skipping states of size 2 have to be considered, thus $1 + 2 \times 7 = 15$ skipping states. These skipping states are presented in Figure 1.

⟨ 60, 72 ⟩	⟨ 381, 72 ⟩	⟨ 325, 369 ⟩
⟨ 366, 72 ⟩	⟨ 381, 368 ⟩	⟨ 359, 72 ⟩
⟨ 366, 60 ⟩	⟨ 369, 72 ⟩	⟨ 359, 325 ⟩
⟨ 368, 72 ⟩	⟨ 369, 381 ⟩	⟨ 55, 72 ⟩
⟨ 368, 366 ⟩	⟨ 325, 72 ⟩	⟨ 55, 359 ⟩

Fig. 1. Example of skipping states obtained using the enhanced skipping variant for $n = 3$

Then, matching trigrams are searched in the model. The corresponding entries are weighted and included in the final recommendation list according to Equation (8).

The following sections study the performance of the model presented in this section.

4. Experimental setup

4.1 Corpus

Empirical studies are performed on two types of datasets. The first one is provided by the Crédit Agricole S.A. banking group[1], one of the main banks in France. Its employees use an Intranet interface containing workspaces, news, articles, etc. The bank provided us anonymized navigation client logs containing 3,391 distinct web pages (resources) browsed by 815 bank clerks during years 2007 and 2008. Using these logs we could extract a corpus of 123,470 consultations.
The second corpus is the CTI web server corpus of the DePaul University (http://www.cs.depaul.edu). It contains 69,471 consultations of 683 pages by 5,446 users during a two week period in April 2002 (*i.e.* about 170 consultations per day). The data has been cleaned and filtered by eliminating sessions of size 1 and low support page views.
The repartition of session sizes of both corpora are depicted in Figure 2. As can be seen, most of the sessions have a size below 10. The Crédit Agricole S.A. corpus has an average session size of 8.33 while the DePaul corpus has an average session size of 5.05.
In order to test the robustness to noise of our model, an increasing percentage of resources is randomly included in the corpora. These resources are extracted from the set of distinct

[1] Thanks to Jean-Philippe Blanchard

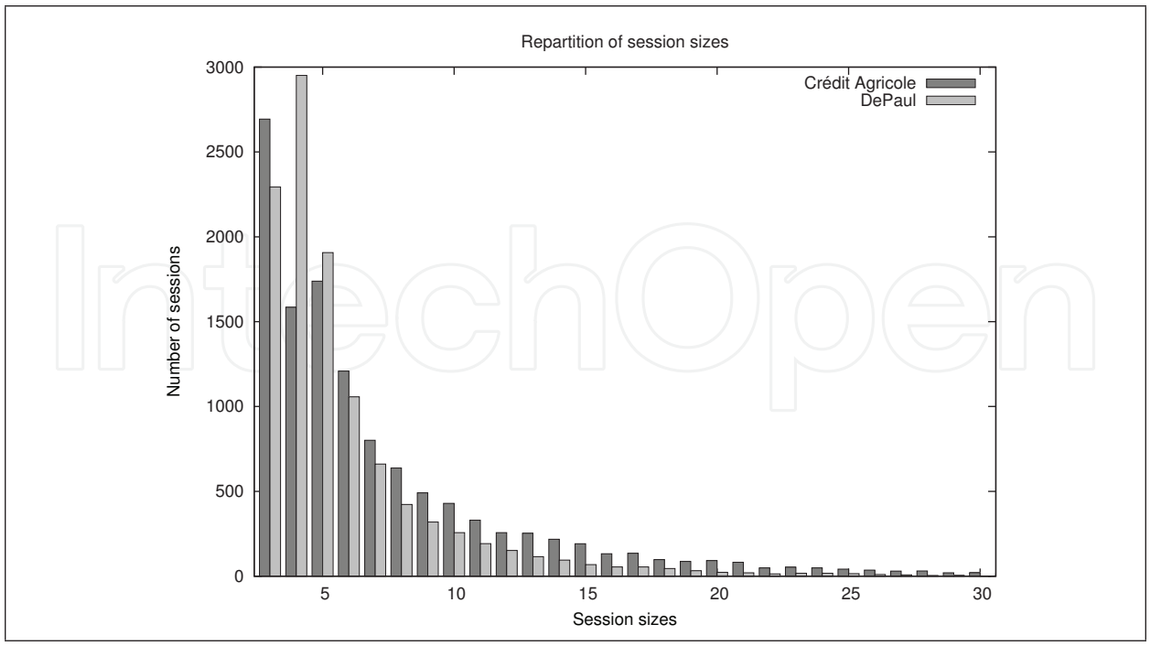

Fig. 2. Repartition of session sizes

resources (the vocabulary) of each corpus. Notice that noise was already present in the original corpora, but was not quantifiable.
Two extra processings have been applied to the data. The first is the elimination of sessions of size 1 (which was already performed on the DePaul corpus) and 2. This is because in the following, we compare our SBR model for $n = 3$ to state-of-the-art models, and wanted the same recommendation cases to be considered. The second is the division of the resulting corpora into training and test sets of 90% and 10% respectively.

4.2 Evaluation metrics

To evaluate the accuracy of our models, we used the Recommendation Score (RS). This metric evaluates the average pertinence of recommendation lists. For each history of the test corpus, a recommendation list of size m is built, containing the most probable resources according to the model. If the actual resource is in the list, the recommendation is pertinent (also called a hit). This metric thus calculates the percentage of pertinent recommendations; it is also called the hit-ratio (Jin et al., 2005; Pavlov et al., 2004). To complete the evaluations, we also provide the coverage, *i.e.* the percentage of cases where the model can recommend a resource. Running times and model sizes are provided too. All experiments have been performed on a 2.66GHz processor and 4GB memory computer. Running times have been obtained by running ten times each model and retaining the smallest ones.

4.3 Models

In the following, we compare our SBR model to state-of-the-art web predictive models. These models are a standard form of all-k^{th}-order Markov model and open sequential patterns.
The sequential patterns are built and selected according to the GSP algorithm and the combination of the antecedents is performed using the maximum length policy as done in (Nakagawa & Mobasher, 2003) together with the sum policy.

Recall that all-k^{th}-order Markov models are equivalent to closed sequential patterns. The only difference is that there is no need of the Apriori or GSP algorithm to filter the states.

5. Tractability

This section is dedicated to a theoretical and an empirical study of the tractability of the SBR model, the all-k^{th}-order Markov model and open sequential patterns.
Two aspects are considered: time and space complexity. As training can be performed off-line, only the time complexity of the recommendation step is studied.

5.1 Theoretical discussion

Theoretically, space complexity is dependent on the number of distinct elements N of the data considered. For instance, if a model has to store all encountered sequences of sizes 1 and 2, then the maximum number of elements to be stored is $N + N^2$.
Time complexity is dependent on the number of sequences that are considered in the history for each recommendation, and the time necessary to find a matching antecedent or state in the model.

5.1.1 Sequential patterns

- Space complexity:

Using sequential patterns, a huge number of sequences has to be stored. If N is the number of distinct elements and D the window size, the maximum number of elements to store is:

$$\sum_{k=1}^{D} N^k = N \cdot \frac{1 - N^D}{1 - N} = \mathcal{O}(N^D) \tag{9}$$

Using the GSP algorithm reduces space complexity; however it induces a lower coverage of longer patterns.

- Time complexity:

Open sequential patterns consider variable length open sequences in a window of size D. The last element of the pattern (the consequent) is always the rightmost element in the window. The number of combinations induced is thus:

$$\sum_{k=1}^{D-1} C_{D-1}^{k} = 2^{D-1} \tag{10}$$

The search of the corresponding patterns in the model, can be done in $\mathcal{O}(k)$ using a tree structure, where k is the length of the current pattern to be matched. The number of iterations of each recommendation is thus:

$$\begin{aligned} \sum_{k=1}^{D-1} \mathcal{O}(k) \cdot C_{D-1}^{k} &\leqslant \sum_{k=1}^{D-1} \mathcal{O}(D) \cdot C_{D-1}^{k} \\ &\leqslant \mathcal{O}(D) \sum_{k=1}^{D-1} C_{D-1}^{k} \\ &\leqslant \mathcal{O}(D) \cdot 2^{D-1} = \mathcal{O}(D \cdot 2^{D-1}) \end{aligned} \tag{11}$$

5.1.2 All-k^{th}-order Markov models

- Space complexity:

The maximum number of elements induced using an all-k^{th}-order Markov model is the same as the one of sequential patterns. However, in practice considering contiguous patterns induces far less elements, and space complexity is lower. The difference depends on the size of the training data and on the number of distinct resources. As for sequential patterns, pruning the states can lower space complexity, as done in (Deshpande & Karypis, 2004), but may also induce a low coverage of longer sequences.

- Time complexity:

All-k^{th}-order Markov models have a lower complexity than sequential patterns. Indeed, as the patterns considered are contiguous, only D sequences are induced for each recommendation. Time complexity is thus:

$$\sum_{k=1}^{D} \mathcal{O}(k) = \mathcal{O}(D^2) \tag{12}$$

5.1.3 SBR model

- Space complexity:

The maximum number of elements induced using the SBR model is always lower than N^n, which is a quite lower upper bound than the ones of both previous models.

- Time complexity:

The complexity of our model depends on the skipping variant used. Using the full skipping variant, C_{D-1}^{k} sequences are induced for each recommendation, thus $\mathcal{O}(D^{n-1})$ if $n \leqslant \frac{D}{2}$, $\mathcal{O}(D^{D-n+1})$ if $\frac{D}{2} \leqslant n \leqslant D$, and $\mathcal{O}(D^{D/2})$ $\forall n \leqslant D$. As searching the states in a tree structure can be done in $\mathcal{O}(n)$, time complextity is thus:

$$\begin{array}{ll} \mathcal{O}(n \cdot D^{n-1}) & \text{if } n \leqslant \frac{D}{2} \\ \mathcal{O}(n \cdot D^{D-n+1}) & \text{if } \frac{D}{2} \leqslant n \leqslant D \\ \mathcal{O}(n \cdot D^{D/2}) & \forall n \leqslant D \end{array} \tag{13}$$

Shani's and the enhanced skipping variants reduce this number to $\mathcal{O}(n \cdot D)$.
Thus, depending on the value of n, the full skipping variant can have a high complexity. However, using low values of n such as 3 or 4 leads to acceptable complexities. For $n = 3$ and $D \geqslant 6$, the time complexity of the full skipping variant is $\mathcal{O}(D^2)$. As well, for $n = 4$ and $D \geqslant 8$, the time complexity of the full skipping variant is $\mathcal{O}(D^3)$. As using skipping allows to simulate a higher order model using a lower order model, the accuracy and coverage should be high.

5.2 Empirical comparison

In this section, experimental results of the three models are compared in terms of model sizes and computation time. In order to have comparable models, all support and confidence thresholds are set to 0. The size of the recommendation lists is set to 10. We chose this value for two reasons: (1) a user rarely takes into consideration resources recommended above this value (2) top-10 recommendation lists are widely used, which provides a direct comparison of the results. The size of the window is set to $D = 10$. The SBR is tested for a value of $n = 3$,

and using the three aforementionned skipping variants: Shani's, enhanced and full. Results are shown in Table 1.

	DePaul		Crédit Agricole S.A.	
	size	time	size	time
SBR (Shani)	1.5 MB	14s	2.3 MB	3m05s
SBR (Enhanced)	2.6 MB	18s	4.1 MB	3m20s
SBR (Full)	5.3 MB	25s	8.1 MB	5m51s
AKO	3.3 MB	3m06s	8.3 MB	17m02s
SP	108.7 MB	1m08s	289.6 MB	10m50s

Table 1. Size and running time of the models

- Space requirements

We can first notice that using the SBR model with Shani's and the enhanced skipping variants provides the lowest model sizes on both corpora. On the Crédit Agricole S.A. corpus, the full skipping variant induces a larger model than the all-k^{th}-order Markov model. However, on the DePaul corpus, the size of the SBR model with the full skipping variant is slightly smaller than the one of the all-k^{th}-order Markov model.

The huge space complexity of sequential patterns is obviously verified: it is more than 20 times larger than all other models on the Crédit Agricole S.A. corpus, and more than 30 times larger on the DePaul corpus. So far, the SBR model and the all-k^{th}-order Markov model are almost equivalent.

- Running time

Surprisingly, the sequential patterns model ran faster than the all-k^{th}-order Markov model (1m08s *vs* 3m06s and 10m50s *vs* 17m05s). This is because the first one considers open sequences and contains far more elements (108.7 MB *vs* 3.3 MB and 289.6 MB *vs* 8.3 MB). Thus using sequential patterns, it is much more likely to find matching sequences and the model is able to provide top-10 recommendation lists after far less iterations. Indeed, the sequential patterns model we implemented use the maximum length policy. Using this policy, for each possible sequential pattern length, all combinations are considered in the window. If a sufficient number of recommendations is induced, then it is not necessary to continue the process using smaller sequential patterns. The same strategy is used for the all-k^{th}-order Markov model.

The running time of the SBR model is clearly below the ones of both other models. Using the full skipping variant, it ran more than four times faster than the sequential patterns on the DePaul corpus, and almost twice faster on the Crédit Agricole S.A. corpus. It thus represents the most tractable alternative. As could have been predicted, the most tractable skipping variants are Shani's and the enhanced skipping variants.

6. Robustness to noise

As discussed previously, the presence of noise in navigations can have dramatic effects on the recommendations. Our model is designed to be robust to noise. In this section we compare its features with the ones of all-k^{th}-order Markov models and open sequential patterns.

6.1 Theoretical discussion

In (Jianyong et al., 2007), it is argued that closed sequences are more appropriate for web navigation. The reasons put forward are that it provides more compact recommendation lists and that it is more efficient. Moreover, all-k^{th}-order Markov models are considered as being among the best performing models of the state-of-the-art. However, using closed sequences makes it impossible to ignore resources corresponding to noise. When the history does not match the model, it is then reduced step by step, until a resource can be recommended. After a reduction, the resource that is discarded is the one that is the more distant from the resource to recommend. So if any noise appears in a close past and no matching history can be found unless this resource is ignored, the history will be reduced until the resource is out of it. As a result, very few resources will be considered to compute the recommendations. Moreover, when the resource previously consulted is noise, no reliable recommendation can be provided. For these reasons, we think that using closed sequences, in particular all-k^{th}-order Markov models, is not the most appropriate configuration.

Open sequential patterns exhibits good characteristics that make them more robust to such problems. As all (2^{D-1}) possible open sequences in the past can be considered, if noise occurred in a recent past, longer sequences that does not include it can be considered to compute recommendations. It should be noticed that using such a scheme, most of the sequences induced are formed using distant resources. We think that such sequences may be less representative, as users rarely perform navigation mistakes, returns to previous pages or parallel navigations between each page consultation, and that most of contiguous consultations correspond to coherent transitions. Moreover, as the number of sequences induced is huge, it is not clear whether it is compensated by the accuracy provided.

The SBR model has several advantages concerning robustness to noise. First, all skipping states used to provide the recommendations are combined, and weighting schemes are used to alleviate the importance of distant resources. Moreover, using Shani's and the enhanced skipping variants, among the n elements of each n-gram, $n-1$ elements are always contiguous, which lowers the phenomenon of non coherent transitions. Last, it has low space and state complexities. It thus represents an even better candidate.

6.2 Empirical comparison

We are now interested in the empirical study of the robustness to noise of the models. Tests are performed on the Crédit Agricole S.A. and DePaul corpora in which 0%, 15% and 30% of noise is inserted. It should be noticed that when no noise is inserted, there is actually already some natural noise in the corpus. Thus, the 0% noise values below does not mean that there is no noise in the corpus, but that no additional noise was inserted. For this reason, we only inserted a maximum of 30% of noise.

We first focus on the determination of the best configuration of our SBR model. We then provide a comparison of the SBR model and the models of the state-of-the-art. Here again, the size of the recommendation lists is set to 10. Results are provided in terms of RS and coverage.

6.2.1 Skipping variants

This section is dedicated to the study of the SBR model. The two features studied are the skipping variant presented in section 3.1 and the weighting schemes presented in section 3.2. Figure 3 and Figure 4 show the RS obtained on the Crédit Agricole S.A. corpus and the DePaul corpus respectively.

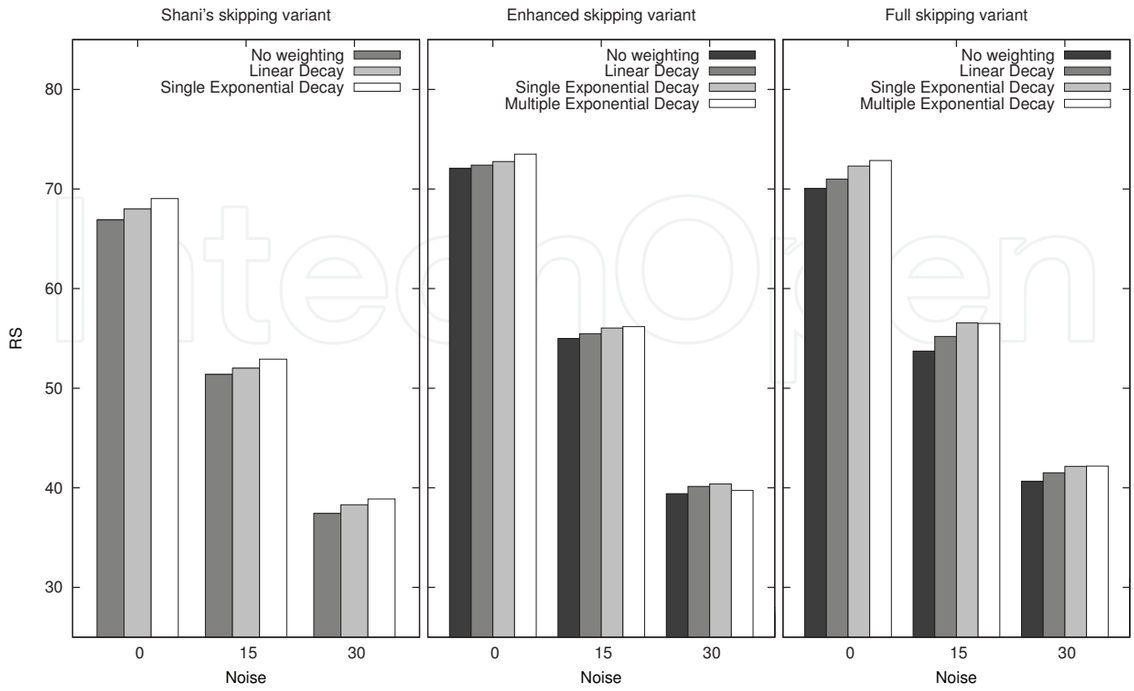

Fig. 3. Accuracy of the SBR model on the Crédit Agricole S.A. corpus according to the skipping variants and the weighting schemes proposed

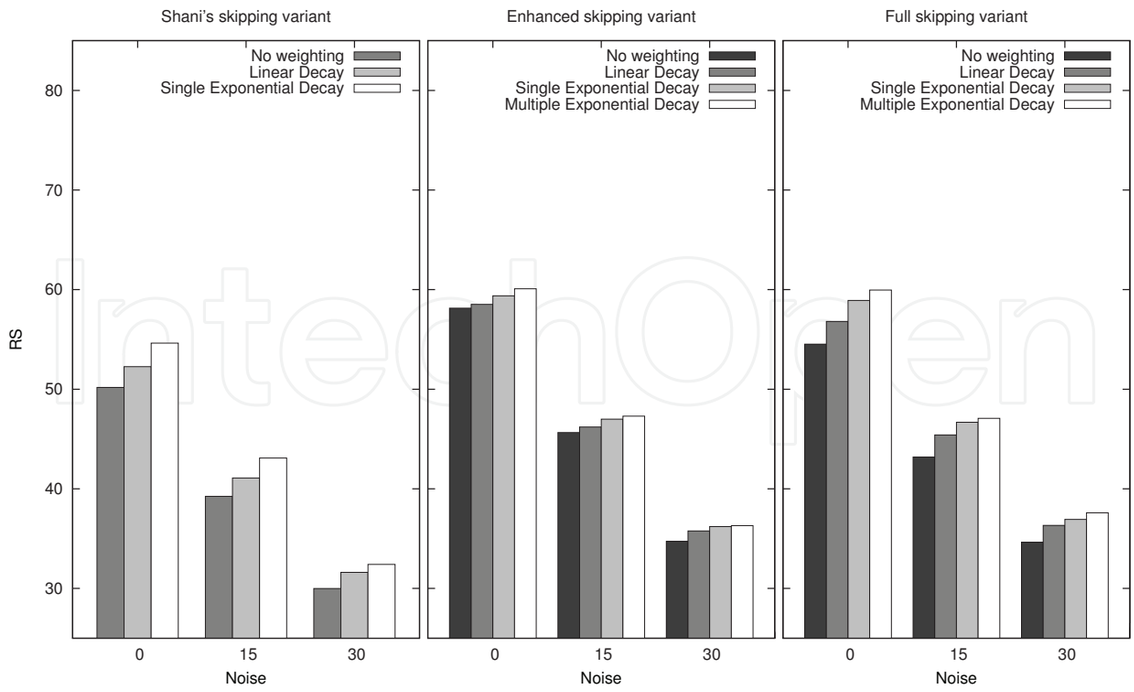

Fig. 4. Accuracy of the SBR model on the DePaul corpus according to the skipping variants and the weighting schemes proposed

We can first notice that Shani's skipping variant provides the lowest RS values on all 6 corpora. When no noise is inserted into the corpora, the enhanced and the full skipping variant provide similar results. When noise is inserted, the full skipping variant provides a slightly better RS. As our enhanced skipping variant is almost as accurate as the full skipping variant and has a lower complexity, it seems to be the best configuration.
Focusing on the weighting schemes, we can first notice that using no weighting provides almost always the lowest RS. When no noise is inserted, the Multiple Exponential Decay weighting scheme always provides the best results. When noise is inserted, it almost always provides the best results. It thus constitutes the best alternative.
So far, the best configuration of the SBR model is the enhanced skipping variant together with the Multiple Exponential Decay weighting scheme.
Focusing on Table 2 and Table 3, we can see that when no noise is inserted, all skippings reach an almost full coverage. When noise is inserted the enhanced and the full skipping variants provide the best coverages, which are similar. This thus confirms that the enhanced skipping variant we proposed is the best configuration.

Noise	0	15	30
Shani	98.8	89.5	84.5
Enhanced	99.5	95.9	93.6
Full	99.7	96.0	94.3

Table 2. Coverage of the SBR model on the Crédit Agricole S.A. corpus according to the skipping variants proposed

Noise	0	15	30
Shani	98.9	96.1	93.1
Enhanced	99.7	98.2	96.3
Full	99.8	98.3	96.4

Table 3. Coverage of the SBR model on the DePaul corpus according to the skipping variants proposed

6.2.2 Comparison to the state-of-the-art

This section is dedicated to the comparison of the robustness to noise of our SBR model to both state-of-the-art models. The configuration of the SBR model is the enhanced skipping variant together with the Multiple Exponential Decay Weighting scheme.
Results of the all-k^{th}-order Markov model and the sequential patterns are provided at optimal pruning thresholds. Notice that sequential patterns could not be used with support thresholds of 0, as moving to more noisy environment made the space requirements too huge for our computer, although it was possible using both other models.
Results are presented in Figure 5 and Figure 6. We can first notice that on the Crédit Agricole S.A. corpus, the best results are provided by the SBR model. When no noise is inserted, the SBR provides a significant enhancement compared to sequential patterns. This difference is

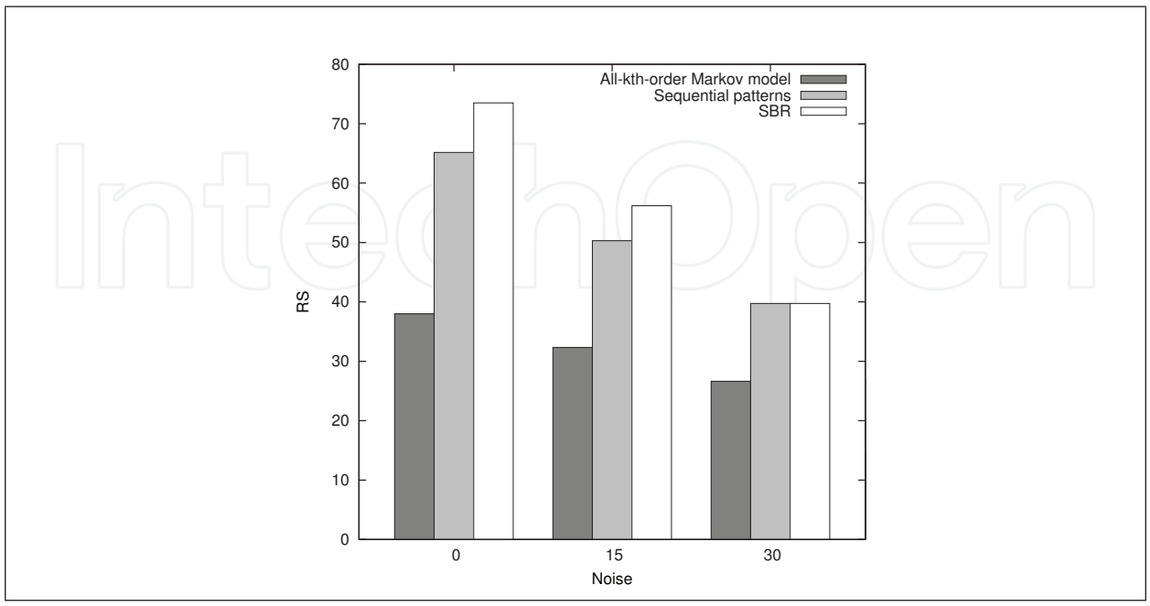

Fig. 5. RS of the models on the noisy Crédit Agricole S.A. corpora

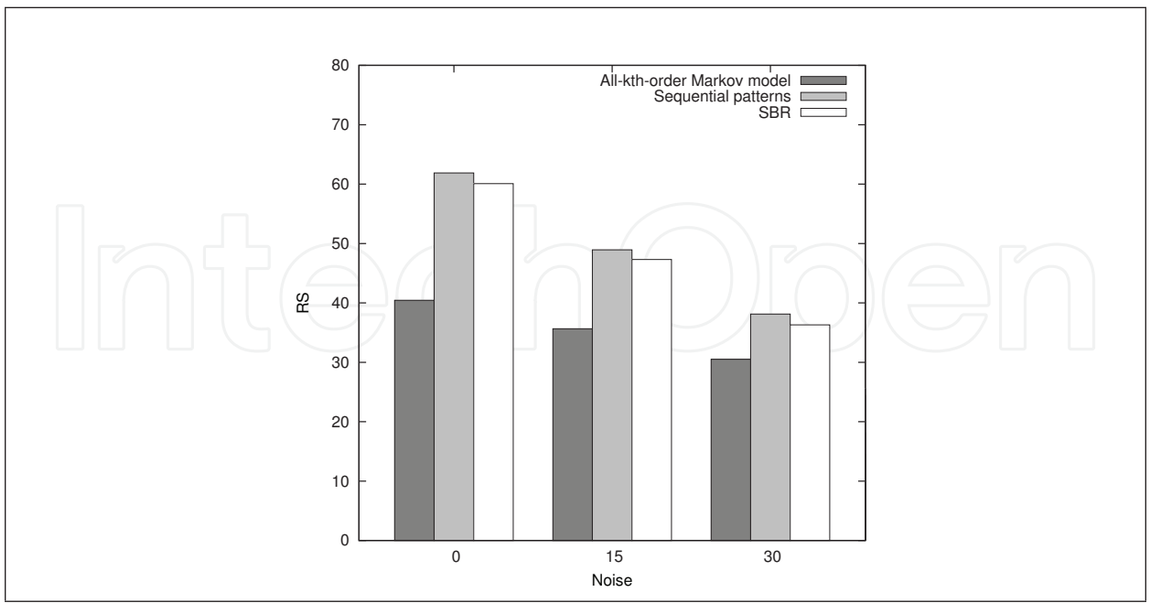

Fig. 6. RS of the models on the noisy DePaul corpora

lower when 15% of noise is inserted in the corpus. With 30% of noise, both models provide similar results.
On the DePaul corpus, sequential patterns provide slightly better results than the SBR model, whatever is the amount of noise inserted. As the SBR has lower time and space complexities, it constitutes a better choice.
The all-k^{th}-order Markov model provides the lowest accuracy, which confirms that using closed sequences provides less accurate results in a noisy environment. It should be noticed that this last model has a higher slope than both other. We think this shows that it is not able to handle long sequences and is more accurate using a lower maximum value of k. Indeed, the more noise is inserted in the model, the less long matching states are found, and thus the lower the length of the matching states. However, as this model provided a very lower accuracy, we did not study this phenomenon further.

7. Conclusion

In this chapter, we focused on sequence-based recommender systems. We first described related work and drew a parallel between natural language and Web navigation. We then decided to take advantage of statistical language models to perform recommendations in the frame of web navigation.
We proposed a new model called Sequence Based Recommender or SBR, that is based on an n-gram model and integrates skipping. This model has the advantage to take into account long histories while being tractable. Several skipping variant were proposed. As well, several weighting schemes were proposed to alleviate the importance of distant resources.
We provided theoretical and empirical studies of the tractability and robustness to noise of our model, compared to state-of-the-art models: all-k^{th}-order Markov models and sequential patterns. The empirical studies were performed on two browsing datasets. Results show that on both corpora, considering open sequences is more efficient than considering closed sequences. Furthermore, our model has been shown to represent the best alternative: it has the lowest time and space complexity, provides a better accuracy on one of the corpora and an accuracy comparable to the one of sequential patterns on the other one, while having a comparable coverage.

8. References

Adomavicius, G., Sankaranarayanan, R., Sen, S. & Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach, *ACM Transactions on Information Systems* **23**(1): 103–145.

Adomavicius, G. & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions, *IEEE Transactions on Knowledge and Data Engineering* **17**(6): 734–749.

Agrawal, R., Imieliński, T. & Swami, A. (1993). Mining Association Rules between Sets of Items in Large Databases, *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pp. 207–216.

Agrawal, R. & Srikant, R. (1995). Mining Sequential Patterns, *ICDE'95: Proceedings of the International Conference on Data Engineering*, pp. 3–14.

Ayres, J., Flannick, J., Gehrke, J. & Yiu, T. (2002). Sequential Pattern Mining Using a Bitmap Representation, *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, New York, NY, USA, pp. 429–435.

Balabanović, M. & Shoham, Y. (1997). Fab: Content-Based, Collaborative Recommendation, *Communications of the ACM* **40**(3): 66–72.

Banko, M. & Brill, E. (2001). Mitigating the paucity-of-data problem: Exploring the effect of training corpus size on classifier performance for natural language processing, *Human Language Technology Conference (HLT)*.

Bonnin, G., Brun, A. & Boyer, A. (2008). Using Skipping for Sequence-Based Collaborative Filtering, *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 775–779.

Borges, J. & Levene, M. (2005). Generating dynamic higher-order markov models in web usage mining.

Boyer, A. & Brun, A. (2007). Natural Language Processing for Usage Based Indexing of Web Resources, *29th European Conference on Information Retrieval (ECIR)*, Vol. 4425 of *Lecture Notes in Computer Science*, Fondazione Ugo Bordoni, Springer Berlin / Heidelberg, Rome, Italy, pp. 517–524.

Chan, S. & Goodman, J. (1998). An Empirical Study of Smoothing Techniques for Language Modeling, *Technical report*, Computer Science Group, Harvard University, Cambridge, Massachusetts.

Chen, Y. & Chan, K. (2003). Extended multi-word trigger pair language model using data mining technique, *IEEE International Conference on Systems, Man and Cybernetics*, pp. 262–267.

Chi, E., Pitkow, J., Mackinlay, J., Pirolli, P., Gossweiler, R. & Card, S. (1998). Visualizing the Evolution of Web Ecologies, *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 400–407.

Das, A., Datar, M., Garg, A. & Rajaram, S. (2007). Google News Personalization: Scalable Online Collaborative Filtering, *WWW'07: Proceedings of the 16th International Conference on World Wide Web* pp. 271–280.

Dempster, A., Laird, N. & Rubin, D. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm, *Journal of the Royal Statistical Society. Series B (Methodological)* **39**(1): 1–38.

Deshpande, M. & Karypis, G. (2004). Selective Markov Models for Predicting Web Page Accesses, *Transactions on Internet Technology* **4**(2): 163–184.

Eirinaki, M. & Vazirgiannis, M. (2007). Web Site Personalization Based on Link Analysis and Navigational Patterns, *Transactions on Internet Technology* **7**(4): 21.

Fleischman, M., Hovy, E. & Echihabi, A. (2003). Offline Strategies for Online Question Answering: Answering Questions Before They Are Asked, *in* E. Hinrichs & D. Roth (eds), *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 1–7.

Goldberg, D., Nichols, D., Oki, B. & Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry, *Communications of the ACM* pp. 61–70.

Goodman, J. (2001). A bit of progress in Language Modeling (Extended Version), *Technical report*.

Huang, X., Alleva, F., Hwang, M. & Rosenfeld, R. (1993). An Overview of the SPHINX-II Speech Recognition System, *HLT '93: Proceedings of the workshop on Human Language Technology*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 81–86.

Jianyong, W., Jiawei, H. & Li, C. (2007). Frequent Closed Sequence Mining without Candidate Maintenance, *IEEE Transactions on Knowledge and Data Engineering* **19**(8): 1042–1056.

Jin, X., Mobasher, B. & Zhou, Y. (2005). A Web Recommendation System Based on Maximum Entropy, *Proceedings of the International Conference on Information Theory: Coding and Computing*, pp. 213–218.

Kim, C. & Kim, J. (2003). A recommendation algorithm using multi-level association rules, *WI '03: Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence*, IEEE Computer Society, Washington, DC, USA, pp. 524–527.

Lu, L., Dunham, M. & Meng, Y. (2005). Mining Significant Usage Patterns from Clickstream Data, *7th International Workshop on Knowledge Discovery on the Web*, pp. 1–17.

Mobasher, B. (2007). *Data Mining for Web Personalization*, LNCS 4321 - Brusilovsky, P. and Kobsa, A. and Nejdl, W., chapter 3, pp. 90–135.

Nakagawa, M. & Mobasher, B. (2003). Impact of Site Characteristics on Recommendation Models Based On Association Rules and Sequental Patterns, *Intelligent Techniques for Web Personalization*.

Och, F. & Ney, H. (2001). Discriminative Training and Maximum Entropy Models for Statistical Machine Translation, *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 295–302.

Park, S.-T., Pennock, D., Madani, O., Good, N. & DeCoste, D. (2006). Naïve Filterbots for Robust Cold-start Recommendations, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, New York, NY, USA, pp. 699–705.

Pavlov, D., Manavoglu, E., Pennock, D. & Giles, C. (2004). Collaborative Filtering with Maximum Entropy, *IEEE Intelligent Systems* **19**(6): 40–48.

Pitkow, J. & Pirolli, P. (1999). Mining Longest Repeating Subsequences to Predict World Wide Web Surfing, *USITS'99: Proceedings of the 2nd conference on USENIX Symposium on Internet Technologies and Systems*, pp. 139–150.

Rabiner, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proceedings of the IEEE*, pp. 257–286.

Rosenfeld, R. (1994). *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*, PhD thesis, Carnegie Mellon University, Computer Science Department.

Rosenfeld, R. (2000). Two Decades of Statistical Language Modeling: Where do we go from here?, *Proceedings of the IEEE* pp. 1270–1278.

Rosenfeld, R. & Huang, X. (1992). Improvement in Stochastic Language Modeling, *Speech and Natural Language*, San Mateo,CA, pp. 107–111.

Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. (2000). Analysis of Recommendation Algorithms for e-commerce, *EC '00: Proceedings of the 2nd ACM conference on Electronic commerce*, ACM, New York, NY, USA, pp. 158–167.

Schechter, S., Krishnan, M. & Smith, M. (1998). Using Path Profiles to Predict HTTP Requests, *Computer Networks and ISDN Systems* **30**(1-7): 457–467.

Schein, A., Popescul, A., Ungar, L. & Pennock, D. (2002). Methods and Metrics for Cold-start Recommendations, *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 253–260.

Shani, G., Heckerman, D. & Brafman, R. (2005). An MDP-Based Recommender System, *JMLR: The Journal of Machine Learning Research* pp. 453–460.

Srikant, R. & Agrawal, R. (1996). Mining Sequential Patterns: Generalizations and Performance Improvements, *EDBT '96: Proceedings of the 5th International Conference on Extending Database Technology*, Springer-Verlag, London, UK, pp. 3–17.

Srivastava, J., Cooley, R., Deshpande, M. & Tan, P. (2000). Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, *SIGKDD Explorations Newsletter* **1**(2): 12–23.

Tan, P. & Kumar, V. (2002). Discovery of Web Robot Sessions Based on their Navigational Patterns, *Data Mining Knowledge Discovery* **6**(1): 9–35.

Trousse, B. (2000). Evaluation of the Prediction Capability of a User Behaviour Mining Approach For Adaptive Web Sites, *In Proceedings of the 6th RIAO Conference - Content-Based Multimedia Information Access*.

Wang, Y., Li, Z. & Zhang, Y. (2005). Mining Sequential Association-Rule for Improving WEB Document Prediction, *ICCIMA '05: Proceedings of the Sixth International Conference on Computational Intelligence and Multimedia Applications*, IEEE Computer Society, Washington, DC, USA, pp. 146–151.

Zhang, Y., Callan, J. & Minka, T. (2002). Novelty and Redundancy Detection in Adaptive Filtering, *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 81–88.

Zimdars, A., Chickering, D. M. & Meek, C. (2001). Using Temporal Data for Making Recommendations., *in* J. S. Breese & D. Koller (eds), *UAI*, Morgan Kaufmann, pp. 580–588.

Web Intelligence and Intelligent Agents

Edited by Zeeshan-Ul-Hassan Usmani

ISBN 978-953-7619-85-5
Hard cover, 486 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

This book presents a unique and diversified collection of research work ranging from controlling the activities in virtual world to optimization of productivity in games, from collaborative recommendations to populate an open computational environment with autonomous hypothetical reasoning, and from dynamic health portal to measuring information quality, correctness, and readability from the web.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Geoffray Bonnin, Armelle Brun and Anne Boyer (2010). Skipping-Based Collaborative Recommendations inspired from Statistical Language Modeling, Web Intelligence and Intelligent Agents, Zeeshan-Ul-Hassan Usmani (Ed.), ISBN: 978-953-7619-85-5, InTech, Available from: http://www.intechopen.com/books/web-intelligence-and-intelligent-agents/skipping-based-collaborative-recommendations-inspired-from-statistical-language-modeling

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen