An Application of Bayesian Networks in Predicting Form Entries

Luigi Troiano*, Gennaro Cirillo†, Cosimo Birtolo† and Roberto Armenise†

* *University of Sannio*
Department of Engineering
82100 Benevento, Italy
troiano@unisannio.it

† *Poste Italiane S.p.A. – TI - SSI*
Centro Ricerca e Sviluppo
80133 Napoli, Italy
{*ciril127,birtoloc,armenis5*}*@posteitaliane.it*

***Abstract*—In this paper we propose a model, based on Bayesian Inference, aimed at predicting the user input according to past interactions. We exploited this approach to prototype an innovative online payment system in Poste Italiane. Preliminary experimental results outline that this approach is feasible and may improve usability and user experience more in general.**

I. Introduction

Auto-completion is regarded as a supportive tool able to greatly simplify the user interaction, especially in e-commerce and e-service domains where filling out information is strongly required. For instance, when we have to buy a book or a plane ticket, application may require to fill out many fields often located on different Web pages.

Software agents help to automate a variety of tasks including those involved in buying and selling products over the Internet. A recent paper surveys several of these agent-mediated electronic commerce systems with special attention to business-to-consumer (B2C) and business-to-business (B2B) aspects [1]. Predicting the next user action is a well known challenge in designing User Interfaces, and auto-completion is one way to address this challenge. Recent researches [2]–[4] suggest that prediction strategies can improve the overall user experience, especially in mobile context where the reduced size of display and keyboard entails a higher user effort. The next action to predict is generally referred to which Web page the user will visit or which character the user will type in filling out a single form field. Word prediction is usually related to single-word completion, although recent works [2] expand this approach to multi-word prediction enabling the auto-completion of whole phrases. Multi-word prediction is helpful in assisting text entry, as it attempts to guess the whole phrase according to words typed so far.

Starting from these results, our work is focused on enhancing user experience by predicting entries in filling out online forms, according to past data entered. So, we are aimed at predicting the whole set of field values in form, instead of single fields in isolation. This leads to time reduction in completing the task, facilitating the user in interacting with digital systems, e.g. kiosks, ATM machines. e-commerce applications, etc. In addition, this facility is able to better support impaired, elder and unskilled users.

In this paper the prediction is performed by a Bayesian Network (BN) as a natural and robust means for modeling the statistical relations among tabular data [5]. Indeed, the model is not required to take into account the sequence of untyped data as in single or multiple word prediction, but to reflect the network of statistical dependencies of labeled non-interchangeable typed data. In other words, modeling sequences of words such as "Robert sends 100 dollars to John" is different from processing a labeled tuple of values such "sender=Robert, receiver=Smith, ammount=100".

The paper describes a prototype of prediction system as starting point for improving user experience in interacting with services given a realistic business environment. The reminder of this paper is organized as follows: in Section 2 we survey some related works; in Section 3 we provide details regarding the prediction model and we briefly outline how the user interacts with the system, describing a case study in Poste Italiane; in Section 4 we report preliminary experimental results, and in Section 5 we present conclusions and future works.

II. Related works

The problem of predicting the next user action has being studied in the area of Human Computer Interaction (HCI) since long time. Davison [6] investigates about user's next action by the analysis of those pages recently requested by the user. Furthermore he overviews different methods to rank URLs in the current page for prediction, such as baseline random ordering, original rank ordering and similarity of the link text with the combined non-HTML text of the

978-1-4244-5612-3/09/$26.00

preceding pages. Zhu, Hong and Hughes [7] use Markov models to predict HTTP requests and Web pages. Low order Markov models seem not to have a good accuracy, as they do not consider sequences of past actions long enough to make a correct guess. Third, fourth or higher order Markov chains are characterized by better accuracy, but lead to higher complexity. Dongshan and Junyi [8] propose a new Markov model to address this issue, called Hybrid-Order-Tree-like Markov Model (HTMM), which makes use of tree-like representation of past user interactions (tree-like Markov model) and an analytical procedure to combine predictions as given by Markov models of different order (hybrid-order).

Berard and Niemeijer [9] demonstrate that word prediction system, specifically PolyPredix$^{\mathrm{TM}}$, provides good results for disabled users in terms of effort reduction. This system provides three different levels of word prediction: word completion, next-word prediction and multi-word prediction and it is based on statistical analysis of word usage patterns. Moreover the authors demonstrate that multi-word prediction is more efficient than next-word prediction or word completion when the prediction system refers to a large amount of past user inputs.

Grabski and Scheffer [3] developed a retrieval model with the task of completing a sentence, given an initial fragment, and given a domain specific document collection. The authors use cosine similarity as a metric to retrieve sentences similar to the query.

Nanopoulos et al. [10] propose prediction schemes aimed at web access prediction. These systems can be useful, for example, in prefetching content before the user request. Moreover, analysis of web usage data leads to discover preferred path within a web site and findings can be used to effectively serve user interaction.

Su et al. [11], classify prediction algorithms in two categories depending on the usage of user interaction log: (i) Point based; (ii) Path based. A point-based algorithm uses the frequencies of past interactions to make a prediction and starts only from the current user insertion, ignoring the previous inputs made during the session. Instead, a path-based algorithm considers all the current interactions of the user, in order to refine the prediction. Studies have demonstrated a point-based algorithm needs fewer information to gain a prediction, but at cost of reliability. Differently, the path-based algorithm uses huge amount of data giving better results. They propose a prediction systems based on *N-gram model*, a typical path-based algorithm. In this model, N-gram is a sequence of length equal to N (i.e. a sequence of words or letters). N-gram model evaluates the percentage of occurrences of the Nth event that follows a sequence of $N-1$ previous events. In order to make a prediction, the algorithm mainly finds, inside of a sequence of N events, frequent K-length sequences derived from $(K-1)$-length sequences, varying the value of K.

Bickel, Haider and Scheffer [12] take on a more language model-based approach, by estimating parameters in linearly interpolated N-gram models.

III. A Prototype of Prediction System

Our work is aimed at developing an intelligent payment system able to support the user in entering data by predicting following entries. Therefore, prediction is not directed to single word auto-completion but to infer following fields values according to fields entered so far. This let the user to focus more on the payment task, to reduce time and effort required to complete the payment, and to eventually correct user mistyping.

The prediction model is based on a Bayesian Network (BN), able to infer future values according to given entries. In particular, each form field is mapped to a node in the network, specifically as discrete random variables whose states represent the values entered in the past. In order to fully specify the BN structure and then to calculate the joint probability distribution, network structure and parameters are learned from data respectively by Necessary Path Condition (NPC) algorithm [13] to find the most relevant connections between nodes and by Expectation Maximization (EM) algorithm [14] to determine conditional probability distributions.

The network is used to predict values given evidence on some fields. The prediction procedure is split in the following 4 steps: (i) Log Storage, (ii) Bayesian Network Learning, (iii) Prediction Making, (iv) Prediction Delivery. The first step consists in logging transaction data when the user fills a form. For instance, if form is made of fields $F_1, \ldots, F_N$, the log will record tuples such as $(f_1, \ldots, f_N)$, where f_i is an admissible value of field F_i, $i = 1..N$. Transaction logs are used as starting point to build a BN as a means to model statistical relations between data on per-user basis (2nd step). This model is used for forecasting fields values according to user entries. We notice that a per-user log is required. Indeed, the prediction model is mostly tailored to specific user history, and privacy policies require not to disclose sensitive information. However we could still learn by user groups as there could exist data correlations in some domains and for some user profiles.

By filling fields, *evidence* is given to the network, and forecasts are performed by inferring the most likely values according to a-posteriori probability on the remaining network nodes (3rd step). More precisely, each unfilled field will be given as *prediction* the state with a-posteriori probability exceeding a given threshold. In general two support modes are possible: the one aimed at automatically filling the entries, the other at suggesting a set of values to the user. The first mode leads to lower user cognitive overhead, but limits the potential prediction power of the system. On opposite, the suggestion list allows to consider multiple choices at a cost of higher cognitive overhead. For this purpose two thresholds can be set, namely $t_1 < t_2$. When the most likely

value has probability over t_2, auto-completion is performed in order to automatically fill a field, although the user can still change the values. Otherwise, most likely values, with probability over t_1 are suggested in a drop-list sorted by decreasing probability, in order to choose a value instead of typing it. Prediction is confirmed by pressing the "Tab" or "Enter" key or by selecting an item in the suggestion list (4th step). A threshold $t_2 > 0.5$ ensures that at most one prediction over the threshold is available at a time. The prediction model has been used as basis for a prototype system in Poste Italiane. The company offers to customers tax and bills payment and money order services, processing a large number of electronic transactions every day. Services are delivered by web, kiosks and ATM machines. Users are demanded to enter payment data regarding for example the sender, the receiver, the amount and the description. The service delivery is basically centered on forms, considered among the most interactive elements of any application. So forms represent a key feature to make applications really useful. Most of payments entail recurrent data with respect to any single user, thus leading to repetitive and sometimes boring tasks.

An example of system usage is provided in Fig.1 and Fig.2.

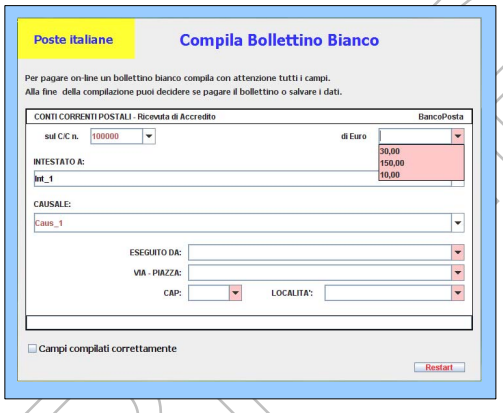

Figure 1. Assisting the completion of on-line money order: Filling the number of checking account and choosing the right import of the money order

In summary the user has three options: (i) to enter directly a field, (ii) to choose in the suggestion list, (iii) to confirm or modify the given suggestion. Therefore the system provides an enhancement as case (i) is the most demanding for the user and the only available option in the current payment systems. Entered data dynamically drive the following prediction. New transactions go to the database in order to feed the user interaction history and to drive the learning process of the prediction model.

IV. Experimentation

The goal of experimentation was to quantify benefits in predicting values based on to user history. We followed a

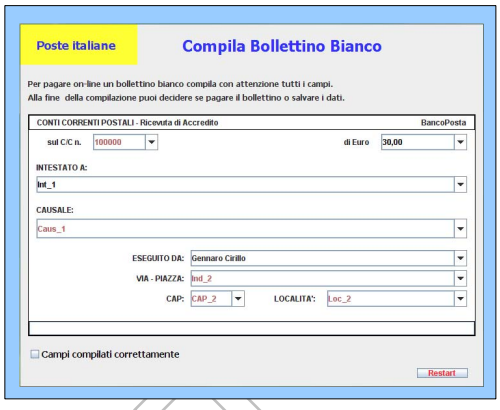

Figure 2. Assisting the completion of on-line money order: Confirmed or filled field in black, while in red predicted ones

double experimentation path. On one side we measured the prediction quality according to some metrics known in literature. On the other we involved a user panel for measuring time to completion saved by adopting our prototype.

A. Experimental Setting

The system has been tested by the following procedure. Let $S = (S_1, S_2, \ldots, S_n)$ be the set of transactions recorded by user interaction in filling out the on-line money order in the past.

For the experimentation, we used a trial dataset made of 60 tuples each one emulates a past user interaction. In particular, every tuple represents data filled out by the user in the bill payment form during online transaction. It is composed by 8 different fields, namely: *Amount* expressed in euros (translated as "*Importo*"), *Number of Checking account* (translated as "*CC*"), *Receiver Name* (translated as "*Intestatario*"), *Description* of the transaction (translated as "*Causale*"), *Sender First Name* and *Sender Last Name* (translated as "*Esecutore*"), *Address* (translated as "*Indirizzo*"), *City* (translated as "*Località*"), *ZIP code* (translated as "*CAP*"). Obviously, in order to simulate a realistic interaction context, we assumed the existence of recurrent user behaviors that lead to the presence of duplicated data.

Prediction quality has been assessed according to the following procedure. A single tuple is randomly chosen in S and it is used for automatically filling out the form. First, one field and one value of it are randomly chosen within test tuple and prediction is performed. Prediction is based on a Bayesian Network previously built. After, a second value is chosen among those fields that have been left or wrongly predicted, and prediction is performed again. The process continues until a certain number of evidences is given and the form is not complete.

In order to evaluate if a field is correctly predicted or not, forecasted value is compared to expected one in the test tuple. This procedure is repeated several times assuming

different values for thresholds. In details, we tested the algorithm providing iteratively two evidences, and assuming threshold $t_2 = \{0.5, 0.6, 0.7, 0.8\}$. Threshold t_1 was not taken into account as we were interested to test the auto-completion feature. The Bayesian Network was obtained by means of structural learning by NPC algorithm at different levels of significance, obtaining networks with 7 and 12 edges, as depicted in Fig.3. Network completeness can affect the quality of predictions, as in general more edges better fit statistical relations among data at higher computational cost. Parameters of each network have been learnt by EM algorithms (iterations = 1000, convergence threshold = 0.0001). Procedure has been iterated 1000 times for each network and test parameter configuration.

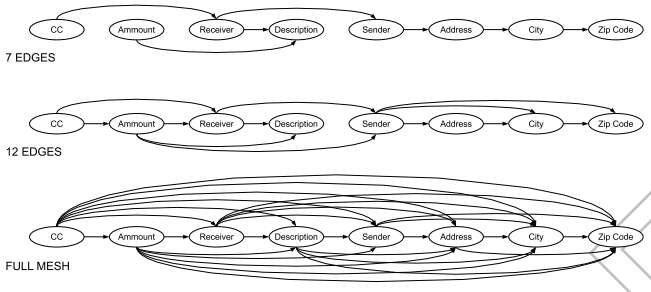

Figure 3. Bayesian networks considered for experimentation

In order to test the network behavior and effectiveness in approximating the correlation among fields, we compared results against a complete (full mesh) network. This represents a theoretical limit, as it is not feasible in reality due to computation overhead required to build the network and to run inference on it.

Moreover, we tested the prediction system with a panel of real users, in order to verify if and how much the prediction improves the user experience. So, we asked 5 users to fill out the money order form and we took note of the time to completion (time necessary to correctly fill out all the fields of the form). Fixed 2 tuples, their task was to fill out the money order form 10 times, 5 with first tuple and 5 with the second one. In the first session test, they were not aided by prediction system, so they were forced to fill out every field. Then the prediction system was activated and users followed the same procedure in 3 different cases: (i) prediction based on full meshed network; (ii) prediction based on a network with 12 edges; (iii) prediction based on a network with 7 edges. During these sessions, they had to fill some fields and to verify if the system correctly predicted the others, manually editing their values if not. All tests have been executed on Pentium IV 2GB RAM.

B. Evaluation Indices

In order to characterize the performance of prediction model we adopted the following metrics [11], [15]: *precision* (Eq.1), *applicability* (Eq.2), and *keystroke savings* (Eq.3).

$$p = \frac{\#\ accepted\ predictions}{\#\ predictions} \tag{1}$$

$$a = \frac{\#\ available\ predictions}{\#\ possible\ predictions} \tag{2}$$

$$s = \frac{\#\ keys_{normal} - \#\ keys_{filled\ out}}{\#\ keys_{normal}} \tag{3}$$

Precision (p) is the number of the good predictions over the total number of predictions, that is a measure of how accurate is the prediction model, similarly to the meaning of this metric in information retrieval literature. Applicability (a) is the number of available predictions over the number of possible predictions, that is the degree of providing predictions, despite they are correct or wrong. Keystroke savings (s) is the percentage reduction in keys pressed compared to fully typed entries: it is a measure of usability of the prediction system, as it provides the theoretical saving[1] of keystrokes, thus the time reduction, in filling out a form.

C. Results

Experimental results are summarized in Fig.4, reporting precision, applicability and keystroke savings (by row) at different network topologies with 7, 12 and full mesh (by columns). Each chart provides the index value at different number of evidences 1,2,3, and for different thresholds (i.e. 0.5,0.6,0.7,0.8). Experimentation outlines the following conclusions.

Using a larger number of edges improves the quality of predictions. Indeed, when the network topology is not able to capture the statistical relations among data performances are in general poor ($L7$), as precision and applicability is on average lower than other cases. In addition, they tend to become worse by increasing the number of evidences, due to a higher number of wrong predictions. However, neither increasing the number of edges over a certain limit (depending on the problem and data characteristics) does not provide any relevant improvement, despite a higher computational cost: over that limit we can notice on average a small decrease of performances, as the model could over-fit data. When the topology is appropriate to fit data relations, we notice an improvement of precision, applicability and keystroke savings by increasing the number of evidences provided. Obviously the highest increment is after the first evidences are provided, as at beginning the network is maximally able to discern among possible predictions.

Precision generally improves by higher threshold values as prediction is expected to be more accurate when a stronger constraint is demanded to prediction. On opposite, applicability decreases by increasing the threshold. Indeed,

[1]It is a theoretical limit as it does not take into account typing errors, control strokes (e.g. Alt, Tab, Ctrl, etc.) and the cognitive overhead in assuming a prediction.

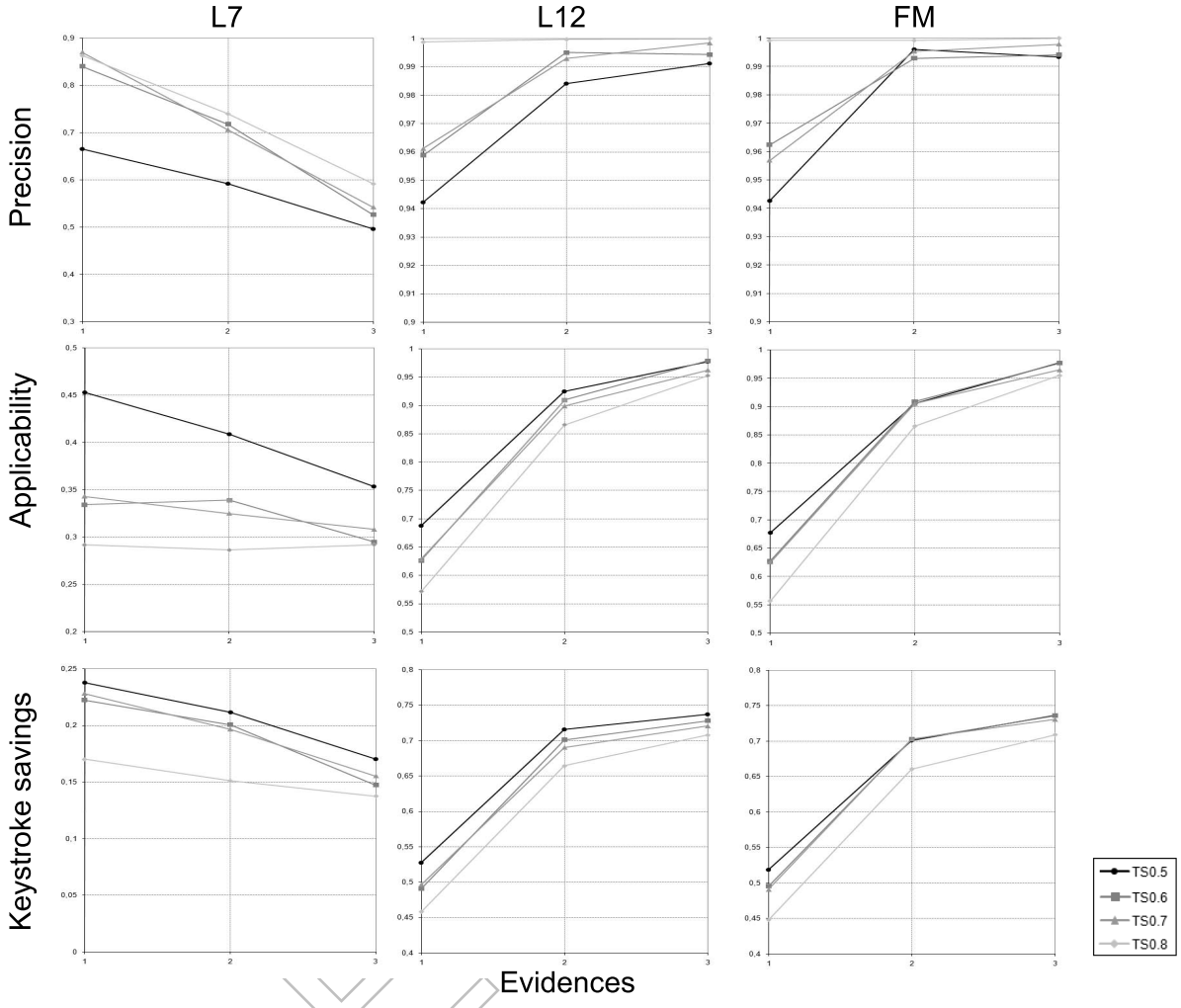

Figure 4. Precision, applicability and keystroke savings

the system is expected to be less able to make predictions with stronger constraints. Obviously the predicted value depends on how recurrent patterns in data are, but they behave consistently even in the case of weaker patterns. This remarks the appropriateness of evaluation indices in capturing the system behavior. Keystroke savings reflects the behavior outlined by precision and applicability: (i) more keystrokes are saved when the precision improves, (ii) stronger constraints limit the ability of making predictions (i.e. applicability is lower) thus keystroke savings is mostly affected by increasing the threshold value.[2] This enforces the opinion that prediction can improve usability.

The very good performances as resulted by experimental results regarding precision, applicability and keystroke savings are due to the system has to predict values already taken into the account in the past. Obviously, depending on the chosen test tuple, it is not said that because of past occurrence, values are always predicted correctly. Indeed prediction depends on a-posteriori probability of values, and this may not exceed the given threshold, depending on frequency of similar patterns in the dataset. Finally, we note that the system prediction power is influenced by the number of network edges. Thus it is clear that best results are obtained using the full meshed network.

On the panel side we collected an average time to completion of 39.96s with no prediction, 14.39s 16.04s and 18.54s with prediction FM, $L12$ and $L7$ respectively. Although we had only a small panel of users, we preliminarily obtained that the prediction makes a difference in filling out a form whether it is accurate (i.e. $L12$, FM) or it is not (i.e. $L7$), and an approximated network (i.e. $L12$) can still provide

[2]This is also related to the definition of keystroke savings as it does not consider the effect of wrong predictions entailing keystrokes for deleting mistakes.

good results comparable to a fully specified model (i.e. FM), as depicted in Table I where p-values of Wilcoxon paired tests between time samples are reported. This result is also confirmed at single user level as reported in Table II with respect to user #1.

	NoP	$L7$	$L12$	FM
NoP	-	$<$ 2.2e-16	$<$ 2.2e-16	$<$ 2.2e-16
$L7$	$<$ 2.2e-16	-	2.532e-02	7.676e-04
$L12$	$<$ 2.2e-16	2.532e-02	-	8.775e-02
FM	$<$ 2.2e-16	7.676e-04	8.775e-02	-

Table I
WILCOXON TEST ON USER TIME TO COMPLETION OF THE WHOLE PANEL, P-VALUES.

	NoP	$L7$	$L12$	FM
NoP	-	5.413e-06	5.413e-06	5.413e-06
$L7$	5.413e-06	-	3.151e-02	5.068e-03
$L12$	5.413e-06	3.151e-02	-	2.179e-01
FM	5.413e-06	5.068e-03	2.179e-01	-

Table II
WILCOXON TEST ON TIME TO COMPLETION OF A SINGLE USER (#1), P-VALUES.

V. CONCLUSION AND FUTURE WORKS

In this paper, we presented an prototype of intelligent system able to assist the user in filling out form by auto-completing and suggesting field values, as means for reducing effort required. In particular the systems refers to ATM and web provision of payment services at Poste Italiane. Prediction is based on statistical inference on Bayesian networks learnt by data. Experimental results show this approach is viable although many issues keep open before making this solution feasible in reality. The main issue regards how to meet real time constraints in delivering the predictions to the user. At the moment, the system provides an answer over the limit of 100ms required for practical applications. Other interesting questions refer to how to consider inter-personal and group information still preserving privacy and how to work with very large database in learning the model.

ACKNOWLEDGMENT

This work was partially supported by MIUR Project Automatic System For The Visually Impaired (SAPI), n.1642-2006.

REFERENCES

[1] M. He, N. Jennings, and H.-F. Leung, "On agent-mediated electronic commerce," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 15, no. 4, pp. 985–1003, July-Aug. 2003.

[2] A. Nandi and H. V. Jagadish, "Effective phrase prediction," in *VLDB '07: Proc. of the 33rd Int. Conf. on Very Large Data Bases*. VLDB Endowment, 2007, pp. 219–230.

[3] K. Grabski and T. Scheffer, "Sentence completion," in *SIGIR '04: Proc. of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 2004, pp. 433–439.

[4] B. D. Davison, "Learning web request patterns," in *Web Dynamics*, M. Levene and A. Poulovassilis, Eds. Springer, 2004, pp. 435–460.

[5] G. E. P. Box and G. C. Tiao, *Bayesian Inference in Statistical Analysis (Wiley Classics Library)*. Wiley-Interscience, April 1992.

[6] B. D. Davison, "Predicting web actions from html content," in *Proc. of the 13th ACM Conference on Hypertext and Hypermedia (HT'02)*, College Park, MD, Jun. 2002, pp. 159–168.

[7] J. Zhu, J. Hong, and J. G. Hughes, "Using markov chains for link prediction in adaptive web sites," in *Soft-Ware 2002: Proc. of the 1st International Conference on Computing in an Imperfect World*. London, UK: Springer-Verlag, 2002, pp. 60–73.

[8] X. Dongshan and S. Junyi, "A new markov model for web access prediction," *Computing in Science and Engg.*, vol. 4, no. 6, pp. 34–39, 2002.

[9] C. Bérard and D. Niemeijer, "Evaluating effort reduction through different word prediction systems," in *SMC (3)*. IEEE, 2004, pp. 2658–2663.

[10] A. Nanopoulos, R. Nanopoulos, D. Katsaros, Y. Manolopoulos, and I. C. Society, "A data mining algorithm for generalized web prefetching," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, pp. 1155–1169, 2002.

[11] Z. Su, Q. Yang, Y. Lu, and H. Zhang, "Whatnext: A prediction system for web requests using n-gram sequence models," *Web Information Systems Engineering, International Conference on*, vol. 1, p. 0214, 2000.

[12] S. Bickel, P. Haider, and T. Scheffer, "Predicting sentences using n-gram language models," in *HLT '05: Proc. of the conf. on Human Language Technology and Empirical Methods in Natural Language Processing*. Morristown, NJ, USA: Association for Computational Linguistics, 2005, pp. 193–200.

[13] H. Steck, "Constraint-based structural learning in bayesian networks using finite data sets," Ph.D. dissertation, Department of Informatics, Technical University Munich, Munich, Germany, May 2001.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[15] D. Bonino, F. Corno, and G. Squillero, "Dynamic prediction of web requests," in *Proc. of the 2003 Congress on Evolutionary Computation CEC2003*, R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, Eds. Canberra: IEEE Press, 8-12 December 2003, pp. 2034–2041.