Machine Learning Based Session Drop Prediction in LTE Networks and its SON Aspects

Bálint Daróczy[†], Péter Vaderna[*], András Benczúr[†]
[*]Ericsson Research, Management & Operation of Complex Systems, Budapest, Hungary
[†]Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary
corresponding author: peter.vaderna@ericsson.com

***Abstract*—Abnormal bearer session release (i.e. bearer session drop) in cellular telecommunication networks may seriously impact the quality of experience of mobile users. The latest mobile technologies enable high granularity real-time reporting of all conditions of individual sessions, which gives rise to use data analytics methods to process and monetize this data for network optimization. One such example for analytics is Machine Learning (ML) to predict session drops well before the end of session. In this paper a novel ML method is presented that is able to predict session drops with higher accuracy than using traditional models. The method is applied and tested on live LTE data offline. The high accuracy predictor can be part of a SON function in order to eliminate the session drops or mitigate their effects.**

Keywords*—Session drop; Machine Learning; Self-Organizing Network (SON)

I. Introduction

Management of Mobile Telecommunication Networks (MTN) is a complex task. Setting up, operation and optimization of MTNs such as those defined by the 3rd Generation Partnership Project (3GPP) need high-level expert knowledge. Therefore it is important for network operators that as many processes in network deployment and operation are automated as possible, thus reducing the cost of operation.

MTNs consist of network elements connected to each other with standard interfaces and communicating via standard protocols. MTNs are managed by Network Management System (NMS) running separately from the network elements. NMS provides functions for network configuration via Configuration Management (CM) and operation supervision via Performance Management (PM) and Fault Management (FM). There are specific functions in the CM, PM and FM systems providing automatic configuration and optimization, usually called self-configuration, self-optimization or self-healing. The common name of these functions in the 3GPP standard is Self-Organizing Network (SON) functions.

The present paper focuses on PM and performance optimization. With the evolution of the generations of the radio and core networks ranging from 2G to 4G, PM reporting functions of the network elements have become higher granularity and more detailed, thus providing better observability. In 2G systems PM relies mostly on counters providing aggregated measurements over a given Reporting Output Period (ROP, usually 15 minutes) within a certain node, in 3G systems it is possible to get higher granularity measurements where per-user events (e.g. Radio Resource Control (RRC) connection setup, sending handover (HO) request, paging, etc.) and periodic per-user measurement reports (sent from the User Equipment (UE) to the nodeB indicating the current radio signal strength and interference conditions) might appear in node logs. In LTE the granularity grows even higher with the possibility of frequent periodic (ROP=1.28 second) measurements per-user and/or per-cell in eNodeBs [1]. Moreover, it is also possible to get the event reports and periodic reports as a data stream, making it possible to process the incoming measurements real-time. The detailed, frequent, high-granularity, real-time reporting enables further processing and analyzing the data and applying them in data-driven techniques to be used in network functions, especially in SON functions. If the SON function is placed at a central network element (e.g. NMS) then the data should be streamed from the eNodeBs. However, if the SON function is placed in the eNodeBs then the PM trace should not necessarily be up and running all the time. In this paper we follow a setup where a Machine Learning (ML) model is trained offline based on detailed node logs and applied real-time where the input can be both from node logs or directly from the internal variables of the node.

In LTE in order to enable communication between the UE and the eNodeB a radio bearer is established. The main metric of interest in the present paper is retainability in LTE systems which is defined as the ability of a UE to retain the bearer once connected, for the desired duration. The release of radio bearers between the UE and the eNodeB can have multiple reasons. There are normal cases such as release due to user inactivity (after expiry of an inactivity timer), release initiated by the user, release due to successful HO to another radio cell or successful Inter Radio Access Technology (IRAT) HO, etc. However, there can be abnormal releases (also called drops) due to e.g. low radio quality either in downlink or uplink direction, transport link problems in the eNodeB, failed HO, etc. Unexpected session drops may seriously impact the quality of experience of mobile users, especially those using real-time services such as Voice-over-IP (VoIP).

The aim of this paper is to introduce and evaluate a ML method to predict session drops before the end of session and investigate how it can be applied in SON. The authors in [2], [3] use traditional ML models, AdaBoost and Support Vector Machine (SVM), to predict call drops in 3G network and use the prediction result to either avoid them or mitigate their effects. The features of the ML model in these studies are aggregated values of certain radio events and reports in a fixed time window preceding the drop. While the settings greatly differ in these studies, the accuracy of our results is much better than in [3] and comparable to [2]. In both papers, prediction is only made where the session is dropped in the

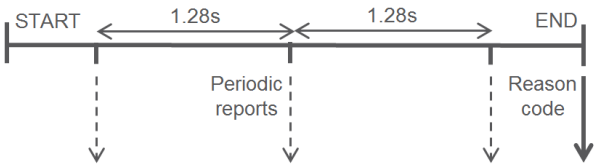

Fig. 1. User session and reporting.

next second. In our paper, we address the SON aspects by evaluating the power of our methods for predicting several seconds before session termination.

We provide an improved ML methodology where the high granularity of the performance reports is exploited and the time evolution of the main features is used as extra information to increase prediction accuracy. We deploy and extend techniques of time series classification [4]. For single parameter series, nearest neighbor classifiers perform the best for time series classification where the distance between two time series is defined by Dynamic Time Warping (DTW) [5]. For session drop prediction, however, we have six simultaneous data sets and hence nearest neighbor methods cannot be directly applied. We define a "similarity kernel" and use Support Vector Machine (SVM) [6] for classification.

II. Network Measurements

A. Constituting user sessions

The analysis is based on raw logs of multiple eNodeBs from a live network containing elementary signaling events indicating e.g. RRC connection setup, UE context release, successful HO to/from the cell, and periodic reports having per-user radio and traffic measurements. The basic unit of information is a Radio Bearer session within a cell. The session is constituted from the elementary signaling events (see Fig. 1). The session is started with setting up an RRC connection or successful HO into the cell from an adjacent cell, and it is ended with a UE context release or successful HO out of the cell. At the end of the session the reason code of the release is reported. Periodic reports are logged during the session every 1.28s containing various radio quality and traffic descriptors.

B. Session records

The essential variables are collected in session records as shown in Table I. There is one variable of interest, the release category that is derived from the release reason code. There are 20 different reason codes, half of them indicating normal release and the other half indicating abnormal release (drop). Each session can have only one reason code. The other variables are contained in the periodic reports and have a time evolution within the session.

The variable to predict is release category that is a binary variable indicating session drop. The variables contributing most to the session drops are selected from a larger set. It contains downlink and uplink parameters. Channel Quality Index (CQI) ranging from 1 to 15 characterizes the quality of the radio channel in downlink direction. Error correction and retransmission mechanisms are operating on different layers of the radio protocols. The retransmission ratio of hybrid automatic repeat request (HARQ) and radio link control (RLC) protocols are reported periodically for both downlink and uplink direction. Signal to Interference plus Noise Ratio on Uplink Shared/Control Channel (*sinr_pusch/sinr_pucch*) characterizes the quality of the uplink shared/control channel. The *sinr_pucch* having a constant value in almost the whole dataset, has been removed from the analysis.

TABLE I. Overview of a Session Record.

Variable	Range	Comment
release category	0 (no drop), 1 (drop)	Derived from the release cause
cqi_avg	1–15	Channel Quality Index
harqnack_dl	0–1	HARQ NACK ratio in downlink
harqnack_ul	0–1	HARQ NACK ratio in uplink
rlc_dl	0–1	RLC NACK ratio in downlink
rlc_ul	0–1	RLC NACK ratio in uplink
sinr_pusch	-4–18	Signal to Interference plus Noise Ratio on Uplink Shared Channel
sinr_pucch	-13–3	Signal to Interference plus Noise Ratio on Uplink Control Channel

C. Time evolution of the variables

The values of the essential variables preceding the end of session have most impact on the release category. However, 1 or 2 seconds before the drop the session is already in a state where the quality is extremely low, making the service unusable. Fig. 2 shows examples for sessions with normal and abnormal release. In the dropped session the *sinr_pusch* decreases and the HARQ NACK ratio increases, indicating uplink problem.

The objective of this paper is to provide a ML method to predict the release category (drop or no-drop) of the session based not only on the features measured directly preceding the end of session but also the time evolution of the features. We consider each session record as a set of time series for the six technical parameters, along with a label of drop or no-drop. For each of the time series, we compute five statistical attributes: minimum, maximum, most frequent item (mode), mean, variance and for each, we compute the gradient. Overall, we obtain a statistical descriptor for a session with 60 attributes: for six time series, we have five statistics and for each, we also have the gradient.

III. Machine Learning Methods

In this section we first give an overview of AdaBoost, our baseline method also used in [2]. Then we describe our new method that aggregates the Dynamic Time Warping time series distance [7] of the six measurement series corresponding to each radio bearer session by an appropriate Support Vector Machine kernel [6]. Finally we describe our evaluation metrics.

A. AdaBoost

AdaBoost [8] is a machine learning meta-algorithm that "boosts" base learners by computing their weighted sum. In each iteration, subsequent weak learners are trained by increasing the importance of the LTE session samples that were misclassified previously.

Our base learners consist of single attributes with a threshold called *decision stump*. For example, a stump can classify sessions with maximum uplink RLC NACK ratio above certain

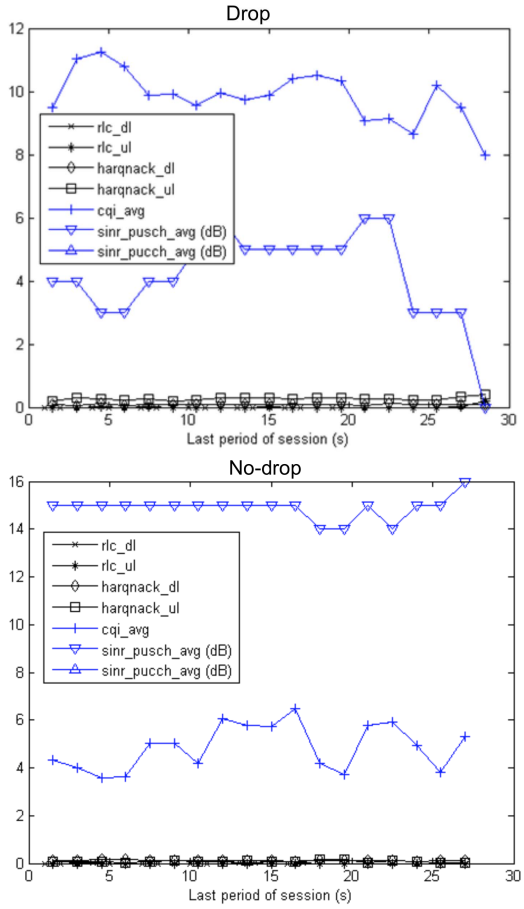

Fig. 2. Typical examples for time evolution of no-drop and drop scenarios.

value as drop, otherwise no drop. We use the AdaBoost implementation of Weka [9] for performing the experiments.

B. Time Series

By an extensive comparative study of time series classification methods [4], the overall best performing time series distance measure is the Euclidean distance of the optimal "dynamic" time warping (DTW) of the two series [5]. Next we define DTW.

Our time series consist of discrete periodic reports. If the length of two series $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_n)$ is identical, we can define their Euclidean distance as

$$L_2(X, Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}. \quad (1)$$

By Dynamic Time Warping (DTW), we may define the distance of series of different length. In addition, DTW warps the series by mapping similar behavior to the same place. For example, peaks and valleys can be matched along the two series and the optimal warping will measure similarity in trends instead of in the actual pairs of measured values. For illustrations of DTW and Euclidean distance, see [5], [4].

The optimal warping is found by dynamic programming. Let the distance of two single-point series be their difference, $\mathrm{DTW}((x_1), (y_1)) = |x_1 - y_1|$. The DTW of longer series is defined recursively as the minimum of warping either one or no endpoint,

$$\begin{aligned} &\mathrm{DTW}^2((x_1, \ldots, x_n), (y_1, \ldots, y_m)) = \quad (2)\\ \min\ &(\mathrm{DTW}^2((x_1, \ldots, x_{n-1}), (y_1, \ldots, y_{m-1})) + (x_n - y_m)^2,\\ &\mathrm{DTW}^2((x_1, \ldots, x_{n-1}), (y_1, \ldots, y_m)),\\ &\mathrm{DTW}^2((x_1, \ldots, x_n), (y_1, \ldots, y_{m-1}))) \end{aligned}$$

C. SVM and our Similarity Kernel

The DTW distance in the previous subsection can be used for classifying time series by any distance based method, e.g. nearest neighbors [4]. In our problem of predicting mobile sessions, however, we have six time series and for a pair of sessions, six distance values need to be aggregated. In addition, we would also like to combine time series similarities with similarity in the statistical features.

A natural idea to handle distances of pairs of observation is to use kernel methods. A kernel acts as an inner product between two observations in certain large dimensional space where Support Vector Machine, a form of a high dimensional linear classifier, can be used to separate the data of dropped and normal sessions [6]. Under certain mathematical conditions, we have a freedom to define the kernel function by giving the formula for each pair of observations.

In order to combine the six distance functions and the statistical features for classification, we use our idea [10] to randomly select a set R of reference sessions and characterize all sessions by their distance from each session in R. For each session s, we obtain $6|R|$ distances from the pairs of the six measurement time series for s and the elements of R. By considering the statistical parameters, we may obtain $|R|$ additional Euclidean distance values between the statistical parameters of s and elements of R, resulting in $7|R|$ distances overall. We use the methods of Jaakkola and Haussler [11] to derive the natural distance metric over the distances. By calculations omitted from this paper, we may obtain that the natural kernel (Fisher kernel) over the $6|R|$ or $7|R|$ distances is the linear kernel over the standard normalized distances. The mean and the variance of the distances can be approximated by the training data.

Over the final feature set, we use SVM for classification [6]. We use LibSVM [12] for training the SVM model.

D. Classification quality measures

The quality of session drop prediction can be evaluated via the class confusion matrix that has four elements corresponding to true drop, true no-drop for the correctly and falsely predicted drop and falsely predicted no-drop for the incorrectly classified sessions. By standard terminology, drop is the positive and no-drop is the negative case and hence we abbreviate the four cases, in order, by TP, TN, FP and FN. As basic metrics, we use the Recall or True Positive Rate = TP/(TP+FN) and False Positive Rate = FP/(TN+FP).

Our main metric for evaluation is derived from the Receiver Operator Curve (ROC) that plots the True Positive Rate as the function of the False Positive Rate by varying the decision threshold. An example ROC curve is shown in Fig. 3. The Area Under the ROC (AUC) [13] is a stable metric to compare

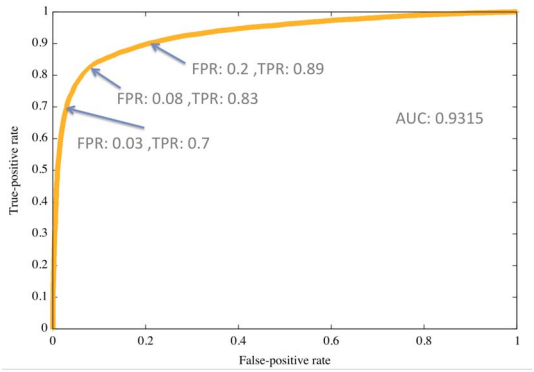

Fig. 3. The Receiver Operator Curve (ROC) of the best predictor for session drop five periodic reports before the actual session drop occurs.

TABLE II. SIZE OF THE SESSION DROP EXPERIMENTAL DATA SET.

	full	no drop, sample	drop
All time series	27.4M	210,000	210,000
At least 15 measurement points	2.8M	23,000	27,440

different ML methods since it does not depend on the decision threshold. As an intuitive interpretation, AUC is the probability that a uniformly selected dropped session is ranked higher in the prediction than a uniformly selected no-drop session.

IV. EXPERIMENTAL RESULTS

Our data consists of 210K dropped and 27.2M normal sessions. To conduct our experiments over the same data for all parameter settings, we consider sessions with at least 15 periodic reports as summarized in Table II. Part of our experiments are conducted over a sample of the normal sessions.

We consider the number of periodic report measurements both before and after the prediction. Data before prediction may constitute in building better descriptors. On the other hand, if we take the last k periodic reports before drop or normal termination, the prediction model is required to look farther ahead in time, hence we expect deterioration in quality. Another parameter of the session is the duration till prediction: very short sessions will have too few data to predict.

Overall, we observe best performance by the DTW based similarity kernel method, followed by the baseline AdaBoost over statistical descriptors. For all practically relevant parameters, DTW improves the accuracy of AdaBoost by at least 5% over the sample as in Table II. Over the full data set, performance is similar: AUC 0.891 for AdaBoost and 0.908 for DTW, with five periodic reports before session termination and at least ten before the prediction. Note that each descriptor needs its own machine learning method: time series with AdaBoost and statistical descriptors with SVM perform poor.

The possible typical physical phenomenon behind drop can be explained by considering the output model parameters. The best features returned by AdaBoost are seen in Table III. We observe that the most important factor is the increased number of packets retransmitted, most importantly over the uplink control channel followed by HARQ over the downlink. Other natural measures as the CQI or even SINR play less role.

In order to see how early the prediction can be made, the performance as the function of the number of periodic reports

TABLE III. BEST FEATURES RETURNED BY ADABOOST.

	drop score	no drop score	weight
rlc_ul max	0.93447	0.12676	1.55
rlc_ul mean	0.11787	-0.11571	0.44 (twice)
harqnack_dl max	0.02061	0.00619	0.29; 0.19
d/dtrlc_ul mean	0.19277	0.18110	0.24
sinr_pusch mean	1.92105	6.61538	0.33

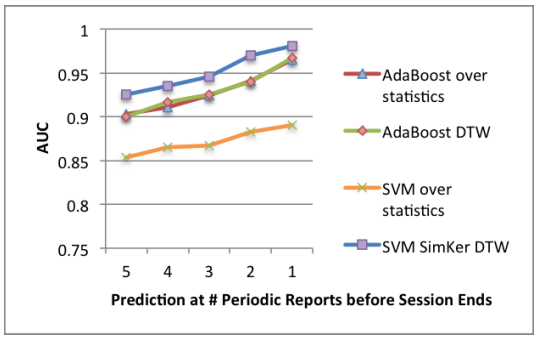

Fig. 4. Performance of early prediction.

before session drop or normal termination is given in Fig. 4. The figure shows the accuracy of early prediction: we observe that we can already with fairly high quality predict drop five measurements, i.e. more than 7 seconds ahead. Regarding the necessary number of observations before prediction, in Fig. 5 we can see that already the first measurement point gives an acceptable level of accuracy. Beyond three reporting periods, most methods saturate and only the DTW based similarity kernel shows additional moderate improvement.

The computational time of feature extraction and the prediction depends linearly on the number parameters of the methods, typically in the range of 1–5 ms per session.

V. SON ASPECTS

For online sessions, several drop avoidance or mitigation mechanisms are proposed in [2], [3], e.g. coordination between voice and data transmissions, switching to other RAT, content pre-fetching, automatic re-connection. However, it is challenging to perform successful actuation on an ongoing session real-time, especially when the session is already in a bad condition. The time for the action may vary (1 - 10 seconds) depending on the complexity of the mechanism. As it is shown in Fig. 4 the accuracy of drop prediction is higher if the model is trained based on the features measured right before the end of session. In a practical perspective of deploying a SON function it is a tradeoff between accuracy and the time left to perform the proper mitigation action.

Every action has its cost due to e.g. increased signaling load, extra delay added to the normal process, misclassification error, etc. On the other hand, if the drop is predicted correctly, the successfully performed mitigation action has a gain. If the costs and gains can be calculated in the same metric, then it is possible to set up a gain matrix to represent the TP, TN, FP and FN cases.

An example gain matrix is shown in Table IV. In this specific model if the prediction result is no-drop and it is correct (TN), then no action is taken and there is no gain

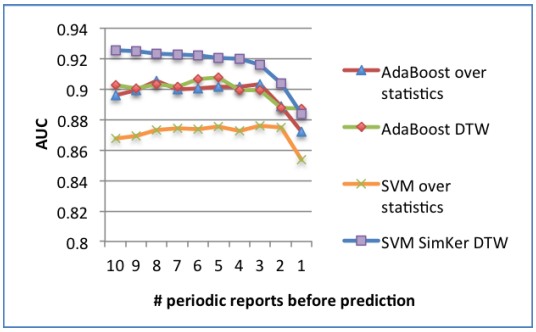

Fig. 5. Dependence of prediction performance on the number of observations.

TABLE IV. GAIN MATRIX OF A SON ACTION.

		Predicted class	
		drop	no-drop
Actual class	drop	+2	-2
	no-drop	-2	0

and no cost. However, if the prediction is no-drop and it is incorrect (FN) or the prediction is drop and it is incorrect (FP) then we have a cost for the incorrect classification (both set to -2 in the matrix). An example for FP case is when the session was unnecessarily forced to a lower RAT where the quality of experience of the user decreased. An example for FN case is when the session was dropped even though the model had predicted another outcome. In case of correctly predicted drop (TP) the overall gain is set to +2.

Once a predictor is up and running it has a certain realization of the FPR and TPR values, indicated by one single point on the ROC curve. Based on the gain matrix it is possible to choose the optimal point of the predictor. Fig. 6 shows the overall gain as the function of the false-positive rate that was calculated from the ROC curve and the example gain matrix. There is a clear optimum point (also indicated in Fig. 3) where FPR=0.08 and TPR=0.83.

The gain matrix essentially expresses the network operator's preference on (TPR, TNR, FPR, FNR) tuple. Knowing the ROC curve of the predictor and the gain matrix the decision threshold of the predictor can be set to an optimal point. For example, if the false positive samples have higher cost then the optimal FPR will be lower.

VI. CONCLUSIONS

High granularity measurements enable more accurate prediction of session drops. A new machine learning approach is shown where not only the aggregated value of the features but the time evolution of the features is taken into account. It is found that in most of the cases the session drop is caused by uplink problem.

The new method outperforms those applied in the literature so far. It is also reported how the accuracy changes if the drops are predicted certain time before the end of session. This enables a SON function using the predictor to balance between how accurate or how early the prediction is performed. It is shown how to optimize the predictor in order to get the highest gain when a mitigation action is applied for sessions about to be dropped.

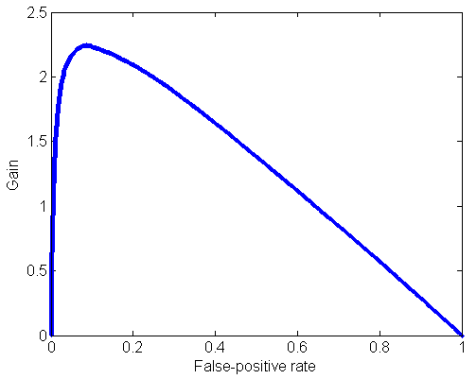

Fig. 6. Gain optimization.

ACKNOWLEDGEMENTS

The first and third authors are supported from the grant OTKA NK 105645, the "Momentum - Big Data" grant of the Hungarian Academy of Sciences and the KTIA_AIK_12-1-2013-0037 project of the Hungarian Government, managed by the National Development Agency, and financed by the Research and Technology Innovation Fund.

REFERENCES

[1] "Subscriber and Equipment (S&E) Trace, 3GPP TS 34.421–423."

[2] S. Zhou, J. Yang, D. Xu, G. Li, Y. Jin, Z. Ge, M. B. Kosseifi, R. Doverspike, Y. Chen, and L. Ying, "Proactive call drop avoidance in umts networks," in *Proc. INFOCOM*. IEEE, 2013, pp. 425–429.

[3] N. Theera-Ampornpunt, S. Bagchi, K. R. Joshi, and R. K. Panta, "Using big data for more dependability: a cellular network tale," in *Proc. 9th Workshop on Hot Topics in Dependable Systems*. ACM, 2013, p. 2.

[4] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," *Proc. VLDB Endowment*, vol. 1, no. 2, pp. 1542–1552, 2008.

[5] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16. Seattle, WA, 1994, pp. 359–370.

[6] B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., *Advances in kernel methods: support vector learning*. Cambridge, MA, USA: MIT Press, 1999.

[7] E. Keogh, "A decade of progress in indexing and mining large time series databases," in *Proc. 32nd international conference on Very large data bases*. VLDB Endowment, 2006, pp. 1268–1268.

[8] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*. Springer, 1995, pp. 23–37.

[9] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed., ser. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, June 2005.

[10] B. Z. Daróczy, D. Siklósi, and A. Benczúr, "Sztaki@ imageclef 2012 photo annotation," in *Working Notes of the ImageCLEF 2012 Workshop at CLEF 2012 Conference*. kn, 2012, pp. 1–6.

[11] T. S. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," *Advances in neural information processing systems*, pp. 487–493, 1999.

[12] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[13] J. Fogarty, R. S. Baker, and S. E. Hudson, "Case studies in the use of roc curve analysis for sensor-based estimates in human computer interaction," in *Proc. of Graphics Interface 2005*, ser. GI '05: Canadian Human-Computer Communications Society, 2005, pp. 129–136.