Data Analysis and Modelling of Users’ Behaviour on the Web

By

Luca Vassio

Supervisor:
Prof. Marco Mellia

Doctoral Examination Committee:
Prof. Dario Rossi, Referee, Telecom ParisTech, Paris (France)
Prof. Antonio Fernandez Anta, Referee, IMDEA Networks, Madrid (Spain)
Prof. Renata Teixeira, Examiner, INRIA, Paris (France)
Prof. Hamed Haddadi, Examiner, Imperial College London, London (UK)
Dr. Alessandro Nordio, Examiner, IEIIT-CNR, Torino (Italy)

Politecnico di Torino
2018
Declaration

I hereby declare that the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Luca Vassio
2018

* This dissertation is presented in partial fulfillment of the requirements for Ph.D. degree in the Graduate School of Politecnico di Torino (ScuDo).
Acknowledgements

When people grow old, they often turn sentimental: in my third thesis, it is finally time to acknowledge someone. First, the institutions that made possible my research: the Italian university system, Politecnico di Torino, Bell Labs and UFMG. From the technical point of view, thanks to Idilio, that more than any other followed me into the dirty side of the work. My supervisor Marco, always helping with his opinions and ideas, also allowing me to stretch my research in different areas. The other professors I had the pleasure to work with, all extremely competent and passionate: Emilio, Carla, Michele, Ana Paula, Flavio and Jussara.

Ringrazio i miei colleghi esperti con cui ho condiviso quotidiane gioie e dolori: Ali, Mori, Martino, Danilo, Leo, Michele, Ryan, Hassan e Enrico. Grazie a Enrico, amico e collega che ha creduto e sudato per la nostra ricerca sull’ottimizzazione meta-euristica.

Un forte ringraziamento ai miei genitori e mia sorella Elena, che mi hanno supportato in tutte le mie scelte, standomi sempre vicino. Ai miei amici di sempre, Dany, Ste, Fra e Sasha, su e per cui potrei scrivere un intero libro. Agli ultimi, intensi e stimolanti mesi con Bianca. Ai miei compagni di serate giochi: Enrico, Marco, Renato, Fabio, Max e Gabri. Alle giornate coi climbers (Ali, Dany, Marco, etc.), ai viaggi in bici e ai miei compagni piloti kartisti. To Kevin, even if he doesn’t speak Italian yet. A tutto il gruppo polacco, con Giordi e Max in prima fila per la loro ospitalità. Un ringraziamento anche ai vecchi matemitici Nick e Marco che tornano spesso nei miei pensieri. E a tutti gli altri amici, tra cui Elena londinese, Dad e Paul.

Muito obrigado aos meus amigos doidos de BH, foram os meses mais incrível da minha vida: os bebados Luis, Matt Marcos J., Matteo, Axxxeli, Brayan e Cristhian; as tranquilas Valeria, Alix, Gabi, Andrea, Raysssa, Thais e Karol. A Evelin, Carola, Dilma, Guillermo, Silvia, Antoine e todo o resto da galera brasileira e internacional. Mulțumesc to Dalia, Marius and Mircea, my super Parisian friends.
Contents

List of Figures xiii

List of Tables xvii

Abstract 1

Abstract - Italian version 3

Abstract - Portuguese version 5

1 Introduction and motivation 7

1.1 Reading map and terminology ......................... 10

1.2 Passive measurements: Tstat ............................ 13

1.3 Datasets and privacy .................................... 14

1.4 Final notes for the reader ............................... 16

2 Detecting user-actions from HTTP traces 19

2.1 Introduction and terminology ............................ 19

2.2 Related work on detecting user-actions .................. 22

2.3 Annotated datasets of volunteers’ activity ................. 23

2.4 User-action classifier design ............................. 25

2.4.1 Feature extraction .................................... 25
2.4.2 Classifier choice ........................................... 27
2.4.3 Performance metrics ...................................... 28

2.5 User-action classifier performance ........................... 29
2.5.1 Feature relevance .......................................... 29
2.5.2 Performance of classifiers and feature sets ................. 29
2.5.3 Testing on smartphone traffic ............................... 31
2.5.4 Training set size ............................................ 32
2.5.5 Comparison with methodology by Ben-Houidi et al. .... 33

2.6 Conclusions and discussion ................................. 34

3 Longitudinal characterization of browsing habits .......... 35
3.1 Introduction .................................................. 35
3.1.1 Scope and limitations ..................................... 38
3.2 Related work on user behaviour and clickstream analyses .... 39
3.3 Datasets: ISP traces ......................................... 41
3.4 Traffic characteristics and impacts on clickstreams detection .... 43
3.4.1 Impact of HTTPS ........................................... 43
3.4.2 Effect of web-page complexity evolution .................... 47
3.4.3 Referer artifacts ............................................ 49
3.4.4 Clickstreams on mobile terminals .......................... 49
3.5 Impact of device on browsing habits ......................... 50
3.5.1 Device usage .............................................. 51
3.5.2 Browsing sessions ......................................... 52
3.5.3 Content consumption ..................................... 56
3.6 The clickstream graph ....................................... 59
3.6.1 Paths characteristics .................................... 59
3.6.2 Connected components ................................... 60
Contents

3.7 Content discovery .............................................. 62
3.8 Conclusions .................................................... 66

4 Users’ fingerprinting techniques from TCP traffic 67
4.1 Introduction and related literature ................................ 67
4.2 Metrics for similarity ............................................ 69
  4.2.1 Jaccard Index .............................................. 69
  4.2.2 Maximum Likelihood Estimation ......................... 70
  4.2.3 Cosine similarity based on TFIDF ........................ 71
  4.2.4 Complexity ............................................... 72
4.3 Identification of Core domains .................................. 72
4.4 Datasets for experiments ........................................ 74
  4.4.1 Ethical and privacy implications .......................... 76
4.5 User fingerprint and identification ............................... 77
  4.5.1 Role of number of domains ............................... 77
  4.5.2 Role of observation time ................................. 78
  4.5.3 Longer profiling or identification? ....................... 80
  4.5.4 Profiling aggregate traffic ............................... 81
  4.5.5 ISP Case .................................................. 82
4.6 Conclusions and discussion ...................................... 82

5 Mining and modeling web trajectories from TCP traffic 83
5.1 Introduction ...................................................... 83
5.2 Trajectories reconstruction and characterization ............... 86
5.3 Modeling user trajectories with TribeFlow .................... 88
5.4 Results ........................................................ 91
  5.4.1 Environments as clusters of common domains ............ 92
5.4.2 Analysis of diversity and popularity of environments .... 93
5.4.3 Transitions in communities ....................... 94
5.5 Conclusions and future work ........................... 95

6 User behaviour in online advertising: modeling and optimization of
temporal contextual ads ................................. 97
6.1 Introduction ........................................... 97
6.2 Related work .......................................... 100
6.3 System model .......................................... 102
6.3.1 Model of user excitation ............................ 103
6.3.2 Model of user response ............................. 104
6.3.3 Timeliness of ads delivery .......................... 107
6.3.4 The SLT strategy and the need of a new approach .... 109
6.4 Analysis under perfect excitation information ............ 112
6.4.1 Arbitrary delivery .................................. 113
6.4.2 Real-time delivery .................................. 117
6.4.3 Buffer-Driven Filtering (BDF) strategy ............... 121
6.4.4 Numerical evaluation ............................... 122
6.4.5 Trace-driven results ................................ 125
6.5 Analysis under partial excitation information ............ 126
6.5.1 Numerical results .................................. 129
6.6 Discussion and conclusions ............................ 130

7 Conclusions and future directions of research ............ 133

Appendix A Exploitation of web navigation data: stakeholders and ethical
issues ...................................................... 137
A.1 Introduction ........................................... 137
A.2 The stakeholders network in the current scenario . . . . . . . . 138
A.3 Ethical and social implications: open questions . . . . . . . . . 143
A.4 Present and future possible alternative scenarios . . . . . . . . . . 144
    A.4.1 Users . . . . . . . . . . . . . . . . . . . . . . . . . . . 144
    A.4.2 Hard laws . . . . . . . . . . . . . . . . . . . . . . . . . . 146
    A.4.3 Researchers and computer scientists . . . . . . . . . . . 148
A.5 Conclusions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 149

Appendix B Papers, collaborations and prizes 151

Bibliography 157
# List of Figures

1.1 Schema of the followed procedure and notes on how it is performed (methodologies and tools). ........................................ 8  
1.2 Tstat is installed at a PoP and monitors the network, logging information from both TCP and HTTP connections. ............. 13  
2.1 Example of a single web-page structure. In this case the main HTML file (orange object) is the URL request of the user-action and the others (yellow objects) are embedded objects, i.e., automatic-actions. 20  
2.2 Example of a client browser activity, where its user-actions are highlighted. User-actions create clickstreams. .................. 21  
2.3 Workflow of the data capture and preparations for obtaining labeled entries with features. ........................................ 24  
2.4 Effects of varying the number of volunteers for training the decision tree. F-measure considered. ............................... 33  
3.1 Incidence of HTTP in the dataset (PoP 1). Fraction of domains in HTTP only, and fraction of their byte volumes. ........... 44  
3.2 Timeline showing when popular domains running on HTTP in 2013 migrated to HTTPS. ............................................ 45  
3.3 Effects of the increasing complexity of web-pages on the percentage of user-actions among HTTP requests. ................. 47  
3.4 Fraction of missing referer for PC browsers and for the anomalous Android Samsung browser in Tablets. .................... 48
3.5 Browser clickstream samples for a PC and a smartphone in an arbitrary day. SEs and OSNs (and their neighbors) are marked in red and blue, respectively. ................................................. 51

3.6 Smartphones present a higher number of browsers, but PCs still dominate the number of user-actions. Browsing from smartphones has increased almost 4 times from 2013. ................................. 53

3.7 ECDF of think-time for browsers. Think-time is shorter on PCs than on smartphones. ................................................................. 54

3.8 Browsing session characteristics. Sessions on smartphones last shorter with fewer web-pages than on PCs. ................................. 55

3.9 ECDF of session idle time. PC browsers access the web more frequently than smartphone ones, but frequency of use is increasing for the latter. ................................................................. 56

3.10 Number of visited web-pages per browser per day. PC browsers visit 3 times more web-pages per day than smartphone ones. ........ 57

3.11 Median number of web-pages and domains visited per day. The number of visited web-pages and domains is typically small, especially for smartphone browsers. Starting months of popular services switching to HTTPS are highlighted. ................................. 58

3.12 Daily number of visited web-pages in the longest path of a click-stream. Depth of the graphs is very limited. ................................. 60

3.13 Box-plots of the number of domains in the paths, per path length. Users stay in very few domains, even when navigating through hundreds of web-pages. ....................................................... 61

3.14 Size of the WCC over time (median over browsers). WCCs typically include SE and OSN. On smartphones, the WCC is bigger than on PCs. This is likely due to the higher usage of SEs and OSNs to reach content. ................................................................. 62

3.15 Fraction of distinct web-pages per clickstream graph that are directly connected to or reachable from SEs and OSNs. Paths starting from SEs include a sequence of multiple web-page visits, while OSNs typically promote a single web-page. ....................................................... 64
3.16 Average number of web-pages directly connected to and reachable from SEs and OSNs. SEs promote more content than OSNs. On smartphones, OSNs have become as important as SEs to directly reach content.  

4.1 Decision tree to identify Core domains.  

4.2 Discovery of domains over time (Campus trace), with two different scales (global and per user).  

4.3 Identification of users considering different number of domains for profiling and identification.  

4.4 CDF of similarity in two consecutive weeks between the same user, or two different users.  

4.5 Accuracy varying the period of identification and profiling on traces.  

5.1 Example of the temporal sequence of Core domains visited by a user, called a user trajectory.  

5.2 Workflow of the followed process for obtaining the trajectories model of the users’ sequential visits on the web.  

5.3 Popularity of domains, considering all of them or just Core ones.  

5.4 Characterization of connections.  

5.5 Each user performs random walks over latent environments, with different probability of interest (preference strength) towards environment.  

5.6 Examples of top-10 domains in different environments of Electronic and Architecture departments, showed as word-clouds. Only TLD+SLD showed for simplicity.  

5.7 Diversity and popularity of the environments for different datasets.  

5.8 Transition probability heat-maps among the environments.  

6.1 High-level view of an advertising system.  

6.2 Example of evolution of user excitation $U(t)$. 
6.3 $P_a$ obtained from (a) Avazu and (b) Outbrain datasets. . . . . . . . . 105
6.4 CTI as a function of delivered impression rate. . . . . . . . . . . . . 112
6.5 Optimal threshold and corresponding CTI as a function of the coefficient of variation of the hyper-exponential distribution of $L$. . . . . . . . 117
6.6 Counterexample showing that the optimal policy for real-time delivery is not always threshold-based. . . . . . . . . . . . . . . . 119
6.7 CTI vs. impression rate $\lambda$ under various strategies, for monotonically decreasing response function $P_a = 0.1e^{-U}$. . . . . . . . . . . . . . . 123
6.8 CTI vs. impression rate $\lambda$ under various strategies, for inverse-U response function $P_a = 0.1ue^{-u}$. . . . . . . . . . . . . . . . . . . . 124
6.9 Performance with different approximations of $U$ in arbitrary and real-time delivery scenarios. Note that plots (a) and (b) depict CTI relative loss and gain, respectively. . . . . . . . . . . . . . . . 131

A.1 The stakeholders network involved in producing, collecting, using and controlling the web navigation data. . . . . . . . . . . . . . . . 139

B.1 Macro-topics of my research and relative papers. . . . . . . . . . . . 153
List of Tables

1.1 Summary of the key terminology and acronyms used throughout the thesis. .......................... 11
1.2 Summary of the passive traces used in the manuscript. ............. 14

2.1 Features and IG with respect to the user-actions. .................. 26
2.2 Performance of the classifiers. F-measure, precision and recall for the user-action class. Cross-validation with PC volunteers’ training set. 30
2.3 F-measure with different features sets with volunteers’ dataset. Cross-validation results shown. ......................... 30
2.4 Performance of the classifiers on smartphone applications. F-measure, precision and recall for the user-action class. Decision tree tested on smartphone traces for different apps. ......................... 32
2.5 Performance of heuristic presented in [1] tested with volunteers dataset and compared to the decision tree. ............... 33

3.1 Summary of the ISP traces. ........................................ 43
3.2 Details per popular browser application of the average number of visited web-pages per day. ............................... 57
3.3 Ranking (top 8 from 2016) of content promoters popularity with the percentage of domains visited from them. .............. 63

4.1 Overview of the considered datasets: traces and characteristics. . . . . 75
4.2 Identified users in two consecutive weeks. .......................... 80
List of Tables

4.3 Identified users in two consecutive days. . . . . . . . . . . . . . . . . 80
4.4 Users correctly identified in residential dataset. . . . . . . . . . . . 82
Abstract

New technologies and services strongly transform our approach with the world. The Internet and its pervasive use was certainly the most dramatic leap in the last 30 years. My research was driven by the need to understand how people interact with the web, capturing its characteristics and changes, and modelling people’s inner habits and interactions. Traces and logs of users’ behaviours collected in the Internet (i.e., passive measurements) offer invaluable information to obtain this goal. Thanks to these passive traces, my work focuses on studying the behaviour of the users on the Internet, with focus on two complementary aspects: (i) data analytics, and (ii) user modelling.

There are many key challenges to face: (big) data requires the use of scalable software and hardware. It demands also the introduction of innovative methodologies and meaningful metric to obtain trustable, filtered, clean and useful information. Data analytics is performed by means of a variety of statistical, machine learning and data mining approaches. Moreover, it is also a pre-requisite for creating analytical models of the studied phenomena, that should be as much as possible adherent to the reality. Lastly, understanding the applicability of derived models is a fundamental step for optimizing performances and understanding possible scenarios.

More in details, during my PhD I analyzed 3 years of data of about 30,000 households. I reconstruct users’ online activity. Thanks to this, I was able to highlight device usage evolution, the intrinsic structure of the navigation and the interactions with social networks and search engines. I introduced a new machine learning approach to identify the intentionally visited web-pages and web-sites. Then, I built specific users’ profiles, fingerprinting their visited domains, and then I showed how to re-identify users in a future time. I modelled the sequence of the visited web services, representing them in a succinct and interpretable manner. I showed that I can automatically extract groups of similar or likely connected web-sites, and
monitor the interests and browsing patterns of single users or communities. I also modelled the user interaction with online recommendation systems, introducing a user behavioural model that captures the impact of the temporal dynamics of shown advertisement. Lastly, I demonstrate how to improve the revenue of an advertisement platform, optimizing the timings when ads are shown to users.

My findings have several direct implications to the different Internet actors and to the research community. Following the scientific approach, I made available the anonymized datasets for the community, in order to guarantee the reproducibility of my results. Moreover, I addressed the problem of privacy online in today changing world, with the objective of finding a trade-off between the desire to obtain knowledge for shaping new technologies and the need to not violate the privacy of individuals. Finally, the current digital transformation implicates that everyone and everything produce data that can be exploited to create new disruptive capabilities. Data analytics allows us to realize incredible transformations not only in the web, but also in our cities, in the energy production, and in manufacturing. Exploiting the knowledge of the users’ behaviour from these data, modelling and optimizing system performances as I did in my work, will be a key factor for designing near future smart-cities.

**Key words:** data analytics, modelling, passive traces, network monitoring, machine learning, human behaviour, fingerprinting, recommendation systems.
Abstract - Italian version

Le nuove tecnologie e le loro applicazioni modificano il nostro approccio con ciò che ci circonda. L’avvento di Internet, con la sua capillarità e pervasività, è stata la trasformazione più importante e repentina degli ultimi 30 anni. La mia ricerca è stata guidata dalla necessità di capire come le persone interagiscano con il web, di catturare come il web stesso cambi, e di modellare le abitudini e i comportamenti degli utenti. Tracce e registri dell’attività online, altrimenti dette misure passive, offrono informazioni inestimabili per raggiungere questi obiettivi. Grazie a queste tracce, il mio lavoro si concentra nello studiare il comportamento delle persone quando navigano su Internet, da due punti di vista complementari: (i) l’analisi dei dati di navigazione e (ii) i modelli analitici di comportamento.

Tuttavia, vi sono molteplici sfide da affrontare: questo tipo di dati, detti big data, necessitano di hardware e software scalabili, e dell’introduzione di metodologie e metriche innovative per ottenere informazioni che siano pulite, affidabili e soprattutto utili. L’analisi dati viene eseguita grazie a metodi statistici, di machine learning e di data mining. Inoltre, l’analisi è un prerequisito per costruire dei modelli analitici dei fenomeni studiati, che siano il più possibile aderenti alla realtà. Infine, capire l’applicabilità dei modelli costruiti è un passaggio fondamentale per ottimizzare le prestazioni e capire i possibili scenari.

Più in dettaglio, durante il mio dottorato, ho analizzato 3 anni di dati di circa 30 000 abitazioni, e ne ho ricostruito le attività online. Grazie a ciò, ho potuto mostrare l’evoluzione nell’utilizzo di diversi dispositivi, la struttura intrinseca delle navigazioni e l’interazione con le reti sociali e i motori di ricerca. Ho introdotto dei sistemi automatici per identificare le pagine e i servizi web intenzionalmente richiesti. Ho anche analizzato la costruzione di profili degli utenti, tracciando i loro domini visitati, per poi mostrare come poterli re-identificare nel futuro. Ho modellato le sequenze di siti visti, rappresentandole succintamente in una maniera
facilmente interpretabile. Ho mostrato come estrarre automaticamente gruppi di siti web simili in contenuto o strettamente relazionati, e come riunire interessi e trend di utenti singoli o intere comunità. Ho anche modellato l’interazione con i sistemi di raccomandazione, introducendo un modello di comportamento umano che cattura l’impatto della dinamica temporale delle pubblicità mostrate. Infine, ho migliorato sperimentalmente i ricavi di una piattaforma di pubblicità, ottimizzandone i tempi di visualizzazione delle inserzioni.

I miei risultati hanno diverse implicazioni per i molteplici attori nel panorama web e per il mondo della ricerca. Seguendo un corretto approccio scientifico, I dataset usati in questa tesi sono resi disponibili in modo anonimizzato per la comunità, in modo da garantire la riproducibilità dei miei risultati. Inoltre, il tema della privacy online in un mondo in forte cambiamento è stato affrontato e analizzato, con l’obiettivo di trovare un compromesso tra il bisogno di ottenere la conoscenza per lo sviluppo delle tecnologie e la necessità di non violare la riservatezza degli individui. Infine, l’attuale trasformazione digitale comporta che tutte le persone e oggetti producano dati che possano essere sfruttati per creare sconvolgenti possibilità. L’analisi dati ci permette di realizzare incredibili trasformazioni non solo di Internet, ma anche nelle nostre città, nella produzione di energia o nell’industria. Sfruttare i comportamenti delle persone che si ottengono attraverso questi dati, modellare e ottimizzare le prestazioni dei sistemi così come ho fatto in questo lavoro, sarà un fattore chiave per progettare le città intelligenti di un futuro molto vicino.

**Parole chiave:** analisi dati, modellazione attività umane, tracce passive, monitoraggio delle reti, apprendimento automatico, comportamento umano, tracciamento online, sistemi di raccomandazione.
Abstract - Portuguese version

As novas tecnologias e as suas aplicações modificaram as nossas interações com o mundo que nos circunda. O advento da Internet, com a sua capilaridade e seu uso generalizado, foi a transformação mais importante e repentina dos últimos 30 anos. Minha pesquisa nasce da necessidade de entender como as pessoas interagem com a web, de compreender como a web está evoluindo, e de modelar os hábitos e comportamentos dos usuários da Internet. Logs que registram o comportamentos dos usuários interagindo com a web, coletados através de medições passivas, oferecem uma oportunidade inigualável para estudar esses fenômenos. Baseado nesse tipo de logs, o meu trabalho foca em dois aspectos complementares: (i) na análise da navegação dos usuários e (ii) na modelagem do comportamento dos usuários.

Muitos desafios devem de ser enfrentados para viabilizar essa análise: medições passivas são em geral volumosas, ou seja big data, e por isso requerem metodologias e infra-estrutura escaláveis para seu processamento. A análise dos dados necessita de métricas significativas e a introdução de metodologias inovadoras para a obtenção de informações confiáveis, filtradas, limpas e, sobretudo, úteis. A análise requer métodos estatísticos, de aprendizagem de máquina e de mineração de dados robustos. Além disso, a análise deve servir de base para a criação de modelos analíticos que sejam aderentes à realidade. Em soma, entender a aplicabilidade dos modelos é um passo fundamental para analisar possíveis cenários de uso e otimizar a performance dos serviços web.

Durante o doutorado eu analisei três anos de dados de cerca de 30 000 consumidores de Internet de alta velocidade, reconstruindo a atividade dos usuários na web. Reconstruí as suas atividades de navegação, destacando a evolução no uso de diferentes dispositivos, a estrutura da navegação e a interação dos usuários com as redes sociais e os motores de busca. Introduzi uma nova metodologia de aprendizado de máquina para identificar páginas web e sites intencionalmente solicitados pelos
usuários nos logs de medidas passivas. A partir dessas informações, demonstrei ser possível criar uma assinatura baseado nos sites visitados por cada usuário, que pode ser utilizadas para re-identificar usuários, com claras implicações para a privacidade on-line.

Modelei a sequência de serviços visitados pelos usuários na web, representando-os de forma sucinta e interpretável. Mostrei como extrair automaticamente grupos de sites similares ou conectados, agrupando os interesses de usuários e de comunidades. Também modelei a interação dos usuários com sistemas de recomendação on-line, apresentando um modelo de comportamento que captura o impacto da dinâmica temporal dos anúncios exibidos nas páginas. Finalmente, mostrei como melhorar os ganhos de uma plataforma de propaganda digital, otimizando os horários nos quais os anúncios deveriam ser exibidos aos usuários.

Os resultados dessa tese têm várias implicações para diferentes personagens na Internet e para a comunidade acadêmica. Na atual transformação digital, todas as pessoas e todos os objetos estão produzindo dados que podem ser explorados para criar novas aplicações revolucionárias. A análise dos dados de navegação nos permite realizar transformações incríveis não só na web, mas também em nossas cidades, na indústria e na produção de energia. Aproveitar o conhecimento do comportamento do usuário obtido a partir de medições na rede e depois modelar e otimizar os sistemas, como feito neste trabalho, será um fator chave para a concepção de futuras cidades inteligentes.

**Palavras-chave:** análise de dados, modelagem humana, medições passivas, monitoramento de rede, aprendizado de máquina, comportamento humano, sistemas de recomendação.
Chapter 1

Introduction and motivation

Nowadays, the web is the preferred means to access almost any kind of information. Online, users read newspapers, buy goods, develop their business, build social relations, etc. Since the origin of the web, understanding how people interact with it has been a fundamental problem for a large variety of purposes like improving search engines [2], increasing online advertisement revenue [3], recommending contents to people [4], or increasing privacy and security [5].

Traces and logs of real users’ behaviours collected in the Internet (i.e., passive measurements) offer invaluable information to understand how people explore the web. Thanks to the availability of huge sets of data, these traces allow verifying (or rejecting) hypotheses and intuitions, extracting significant statistics and tuning analytical models.

However, there are many key challenges to face. These logs need many innovative methodologies in order to obtain trustable, filtered, clean and useful information. New meaningful metrics to characterize the user behaviour evolution must be introduced and analyzed. Moreover, understanding the applicability of derived models (e.g., interaction with advertisements) is a key step for optimizing performances or forecasting future scenarios.

Therefore, my work focuses on two fundamental aspects while studying the users’ behaviour on the web thanks to passive traces: (i) data analytics, and (ii) user modelling.
The different works of this thesis share the same schema, presented in figure 1.1. The starting points are traces, often consisting in raw data, automatically saved in different kind of logs. The passive traces I am studying include TCP packets logs and advertisement platform logs (see section 1.3). Part of my work required also active traces, performed instrumenting browser applications and crawlers.

Many of these passive traces can be classified as big data and therefore requires use of scalable software and hardware: data storage and manipulation has been done thanks to the use of Hadoop and the software Spark on cluster of servers.

Data analytics is performed by means of a variety of statistical, machine learning and data mining approaches. To do that, I was helped by specific software like Weka and Matlab. I also used, in the different steps, specific python libraries for data science like scikit-learn, SciPy and NetworkX. Data analytics is also a pre-requisite for creating an analytical model of the phenomenon under study. In order to be adherent to the reality, a model still requires data to verify the hypothesis and tune its parameters.

The objective of modelling, optimization and data analysis are: (i) understanding real-world phenomena, (ii) forecasting new scenario, and (iii) improving the performance of the systems.

This same approach will be seen to the problems I faced in the different chapters of the thesis.
The most important and broaden research questions and challenges for my thesis are the following:

- How can we process large logs describing years of activity in thousands of households and extract meaningful information from them?
- How is the browsing behaviour of individuals affected by the web evolution over the past years?
- Which are the web-pages and web-sites intentionally requested by users? How can we identify them?
- Can we understand difference in the usage of particular browsing devices (e.g., PCs and smartphones)?
- Is it possible to build a specific user profile fingerprinting his/her visited domains?
- Do these profiles allow re-identifying users in a future time?
- Can we accurately model the sequence of the web services visited by users?
- Is it possible to extract environments with similar or likely connected websites?
- How can we automatically extract the interests of communities of people?
- Can we model the user interactions with online recommendation systems and their temporal dynamics?
- How can we increase the revenue of advertisement platforms? Can we optimize the timings when ads are shown to users?

Going through the thesis, you will find answers to these and other important questions, with the main contributions and implications summarized in chapter 7.

Following the scientific approach, I made available the processed traces for the community, in order to guarantee the reproducibility of my results. I will also discuss the problem of privacy online, with the objective of finding a trade-off between the desire to obtain knowledge for shaping new technologies and the need to not violate the privacy of individuals.
1.1 Reading map and terminology

The thesis is structured as follows. This first introductory chapter is devoted to present the problems I wanted to solve, the challenges I faced in my work, and the datasets and tools I used.

In chapter 2 I study whether machine learning algorithms can successfully be used to detect web-pages intentionally requested by users on HTTP traces. I will show how my approach allows automatic tuning of parameters, how it learns which features are the best candidates for solving the problem, and it could easily be adapted to different scenarios and the evolution of web-pages.

In chapter 3 I thoroughly investigate browsing habits of internauts, providing the evolution over three years. I apply the classifier of chapter 2 to a longitudinal dataset (described in section 3.3) and characterize the browsing habits (section 3.5), clickstreams (section 3.6) and content promoters (section 3.7).

Privacy and user tracking are hot topics that impact everyone who uses the web. Encryption limits access to exchanged information, yet a lot of information can be extracted. In chapter 4 I explore techniques for users’ fingerprinting and identification using only the domains of visited web-services. I show how to extract web-services intentionally requested (section 4.3), that will prove to better characterize users. Results show that my approach can be used to solve the identification problem in different scenarios, if users are profiled for enough time.

Understanding how people move within web-sites is an important problem for a variety of purposes like recommending content, comparing rankings in the web, or increasing privacy and security. Armed with the sequences of visited domains, in chapter 5 I model each user as a random surfer over latent environments. Thanks to the model, I show how to automatically group together the interests and browsing patterns of single users and/or communities.

In chapter 6, I focus on the influence of online advertisements on the users and the interaction with recommendation systems. I introduce a model of an advertising system including a user behavioural model that specifically captures the impact on performance of the history of impressions shown to each user. The proposed model, validated by traces of real advertising systems, allows significantly increasing the revenue of an advertisement platform.
Throughout the work presented in this thesis, I used traces of web navigation data of real users. Appendix A will deepen the subject of the entities that can collect and access these kind of data, other than researchers, highlighting what are the privacy and ethical issues that arise for users, companies, scientists and governments and presenting some of the current legislation.

Appendix B will briefly summarize my papers, collaborations and prized obtained during this PhD.

In chapter 7 I report my main contributions to the research community and the most relevant results obtained thanks to this work. Moreover, I report there the deeper implications to the Internet actors.

The key terminology and acronyms used throughout the thesis are summarized in table 1.1. It contains both common terms and ad-hoc terms I defined. Please refer to this table for doubts in the remaining of the manuscript.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP address</td>
<td>A numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. An IP address serves for host or network interface identification and for location addressing.</td>
</tr>
<tr>
<td>Internet</td>
<td>The global system of interconnected computer networks that use the Internet protocol suite to link devices worldwide.</td>
</tr>
<tr>
<td>DNS</td>
<td>The Domain Name System is a hierarchical decentralized naming system for computers or other resources connected to the Internet. It translates domain names to the IP addresses needed for locating and identifying computer services.</td>
</tr>
<tr>
<td>Web</td>
<td>Informal for World Wide Web (WWW). It is an information space where documents and other resources are identified by URLs, possibly interlinked by hypertext links, and can be accessed via the Internet.</td>
</tr>
<tr>
<td>HTTP</td>
<td>The Hypertext Transfer Protocol is an application protocol for distributed, collaborative, and hypermedia information systems. It represents the foundation of data communication for the web.</td>
</tr>
<tr>
<td>HTTPS</td>
<td>It is an adaptation of the HTTP for secure communication over a computer network, where the communication protocol is encrypted.</td>
</tr>
<tr>
<td>FQDN</td>
<td>A Fully Qualified Domain Name is a domain name that is completely specified with all labels in the hierarchy of the DNS, having no parts omitted (e.g., <a href="http://www.example.com">www.example.com</a>).</td>
</tr>
<tr>
<td>Top level domain</td>
<td>It is the last label of a fully qualified domain name in the DNS hierarchy. For example, in the domain name <a href="http://www.example.com">www.example.com</a>, the top-level domain is com.</td>
</tr>
<tr>
<td><strong>Second level domain</strong></td>
<td>In the DNS hierarchy, it is the label directly below a top-level domain. For example, in example.com, example is the second-level domain.</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Domain</strong></td>
<td>In chapter 2 and 3 it will be used for the pair formed by the second-level domain and the top-level domain of a URL (e.g., example.com), while in chapter 4 and 5 domain will be used for identifying the whole FQDN. Informally, a domain is a web-site.</td>
</tr>
<tr>
<td><strong>URL</strong></td>
<td>A Uniform Resource Locator is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it. E.g., <a href="http://www.example.com/index.html">http://www.example.com/index.html</a> is a URL, which indicates a protocol (HTTP), a FQDN (<a href="http://www.example.com">www.example.com</a>), and a file name (index.html).</td>
</tr>
<tr>
<td><strong>User-action</strong></td>
<td>The initial HTTP request triggered by a user interaction with a browser. Informally, a user-action is a click or a visit to a web-page.</td>
</tr>
<tr>
<td><strong>Web-page</strong></td>
<td>A URL without parameters (e.g., <a href="http://www.example.com/index.html">http://www.example.com/index.html</a>). Many user-actions thus can be related to the same web-page.</td>
</tr>
<tr>
<td><strong>Referer</strong></td>
<td>The HTTP header that identifies the URL of the web-page that linked to the resource being requested.</td>
</tr>
<tr>
<td><strong>Browser application</strong></td>
<td>An application used to navigate through multiple web-pages and domains. E.g., Firefox is a browser application.</td>
</tr>
<tr>
<td><strong>Browser</strong></td>
<td>A specific version of a browser application, identified by its user agent string, used in a specific <em>household</em>.</td>
</tr>
<tr>
<td><strong>Clickstream</strong></td>
<td>An ordered list of user-actions from a browser that can be represented through a graph.</td>
</tr>
<tr>
<td><strong>ISP</strong></td>
<td>An Internet Service Provider is an organization that provides services for accessing and using the Internet.</td>
</tr>
<tr>
<td><strong>CDN</strong></td>
<td>A Content Delivery Network is a geographically distributed network of proxy servers and data centers to distribute services, providing high availability and performance.</td>
</tr>
<tr>
<td><strong>OSN</strong></td>
<td>An Online Social Network is an Internet based application that people use to build social relations with other people who share similar or career interests, activities, or real-life connections.</td>
</tr>
<tr>
<td><strong>SE</strong></td>
<td>A Search Engine is a software system that is designed to search for information on the web. The search results are generally URLs.</td>
</tr>
<tr>
<td><strong>Core domain</strong></td>
<td>A domain originally contacted to download the main HTML document of a page. intentionally visited by a user. A Core domain is intentionally visited by a user, like <a href="http://www.facebook.com">www.facebook.com</a> and en.wikipedia.org.</td>
</tr>
<tr>
<td><strong>Support domain</strong></td>
<td>A domain contacted by the browser to fetch a object that compose a web-page or contacted by other background applications, i.e. a domain that is not Core, like static.10.fbdn.net and dl-client.dropbox.com.</td>
</tr>
<tr>
<td><strong>Tstat</strong></td>
<td>It is a deep packet inspection tool for network monitoring that exposes information from both TCP and HTTP connections. See section 1.2.</td>
</tr>
</tbody>
</table>
1.2 Passive measurements: Tstat

In my thesis I rely on Tstat [6] to collect data. Tstat is a deep packet inspection tool for network monitoring that logs information from both TCP and HTTP connections (see figure 1.2). It was developed by my University (Politecnico di Torino) and it is freely available at http://tstat.polito.it/. Tstat monitors each TCP connection, exposing information about more than 100 metrics. Tstat also implements DPI mechanisms to identify application layer protocols, such as HTTP and HTTPS. Moreover, Tstat records the server Fully Qualified Domain Name (FQDN) the client resolved via previous DNS queries, using its DN-Hunter plugin [7]. This mechanism allows knowing which FQDN the client contacted when accessing a given server IP address, and tracks the usage of HTTP/HTTPS per domain.

Tstat saves flow-level statistics simultaneously to the collection of HTTP logs. These statistics include the number of exchanged bytes/packets, flow start/end timestamps, and, for each HTTP request/response pair, timestamp, server hostname, client IP address, URL, referer, user agent string, content type, content length, and status code. To reduce the privacy risks, Tstat anonymizes IP addresses and removes parameters from URLs in both GET and in the referer fields.

In this thesis, I used TCP logs captured by Tstat in an European ISP network and in my university campus. For the ISP, three probes running Tstat, have been installed in Points of Presence (PoPs) of different cities, where they observed about 25 000 households overall. Each household is assigned, and uniquely identified by, a static
Table 1.2 Summary of the passive traces used in the manuscript.

<table>
<thead>
<tr>
<th>Data origin</th>
<th>Description</th>
<th>Chapter</th>
<th>Available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tstat HTTP logs/browsers histories</td>
<td>HTTP requests and features</td>
<td>2</td>
<td>Yes</td>
</tr>
<tr>
<td>Tstat ISP HTTP logs</td>
<td>User-action clickstreams</td>
<td>3</td>
<td>Yes</td>
</tr>
<tr>
<td>Tstat University TCP logs</td>
<td>Set of domains per user</td>
<td>4</td>
<td>Yes</td>
</tr>
<tr>
<td>Tstat ISP TCP logs</td>
<td>Set of domains per user</td>
<td>4</td>
<td>Yes</td>
</tr>
<tr>
<td>Tstat University TCP logs</td>
<td>User trajectories</td>
<td>5</td>
<td>Yes</td>
</tr>
<tr>
<td>Avazu advertising platform</td>
<td>Ads impressions and clicks</td>
<td>6</td>
<td>Yes</td>
</tr>
<tr>
<td>Outbrain advertising platform</td>
<td>Ads impressions and clicks</td>
<td>6</td>
<td>Yes</td>
</tr>
</tbody>
</table>

IP address. Users connect to the Internet via ADSL or fiber, using a single access gateway offering Ethernet and WiFi home network. For more details, refer to section 1.3 and each related chapter.

1.3 Datasets and privacy

In this thesis I am using many datasets involving passive traces for extracting and modeling different aspects of the user behaviour on the web. In the rest of this section and in table 1.2 I report a summary of the datasets used in the different chapters. Notice how all the datasets are available online (anonymized): most of the results of this manuscript can therefore be validated, repeated and extended by any external researcher.

In chapter 2, I relied on volunteers to obtain the ground truth of user-actions. I collected the visited web-pages from volunteers’ PCs by extracting their browsing history from three browser applications: Safari, Chrome and Firefox. In total, I observed more than 12,000 visits to more than 2,000 web-sites in 3 months of browsing activity of 10 volunteers. In the same chapter, for validating the user-action methodology for mobile browser applications, i.e., apps that can be used to navigate through multiple web-pages and domains, I collected traces by manually visiting popular HTTP web-sites with different apps in an Android smartphone. I choose four different mobile browser applications: Chrome, UC browser, Instagram and Facebook.
During the same period, my campus network was instrumented to passively collect the raw HTTP logs of these volunteers and of the Android smartphone used for the experiment by observing the traffic flowing in and out of my university. I used Tstat (see section 1.2 for details) to perform the collection, recording more than 0.6 million HTTP requests related to the volunteers.

I made the training and validation data of chapter 2 available at https://bigdata.polito.it/content/longitudinal-dataset-clickstreams, where each HTTP request have a list of features and is annotated with a label indicating whether the HTTP request is a user-action or not.

In chapter 3 I evaluate a long-term dataset of HTTP logs captured in a European ISP network to study how users interact with the web and how such interactions evolve over time. After extracting the requested web-pages and the hyperlinks followed by people, I build clickstream graphs for each browser in a household. In total, I construct 5.5 million graphs corresponding to over 1 billion visited web-pages. I made anonymized clickstreams available for researchers at https://bigdata.polito.it/content/longitudinal-dataset-clickstreams.

In chapters 4 and 5 I performed passive measurements using Tstat in my university Campus in Torino, where I consider the traffic of approximately 2,500 users during 4 weeks between January and February 2017. Moreover, I used Tstat during the same 4 weeks period in the same European ISP of chapter 3, where I restrict to the traffic of \( \approx 5,000 \) ADSL and fiber households, behind which multiple user devices may be connected to the Internet. I contribute anonymized sets of domains and anonymized trajectories of users to the community at http://bigdata.polito.it/content/domains-web-users.

In chapter 6 I used passive traces from two advertisement platforms, Avazu and Outbrain. These datasets are publicly available over the Kaggle platform.\(^1\) The first trace reports the click/no-click actions performed by 9 million mobile users on online ads, over 10 days. To limit the impact of noisy data/outliers, I decided to restrict my attention to similar active users, obtaining a filtered dataset that contains 2 million impressions and 21,000 users. Instead, the Outbrain trace contains click/no-click information on roughly 87 million impressions over a period of 14 days.

It is fundamental to find a trade-off between the desire to obtain knowledge for shaping new technologies and the need to not violate the privacy of individuals.

Both the data collection processes and the collected datasets have been discussed, reviewed and approved by the ethical board of my university. I took all possible actions to protect leakages of private information from users. No information to identify people exists in the datasets. In particular, I anonymized the IP addresses of clients using a technique based on irreversible hash functions, only retaining the data that is strictly needed for my studies. Considering the data collection process in the ISP, the same precautions have been implemented, and the data collection process has been reviewed and approved by the ISP security board. In this second scenario, I have no personal information at all about the ISP customers. ISP home Internet installations are identified by anonymized keys, and browsers by user agent strings. Analyses of chapter 3 are thus performed in a per-browser level – i.e., each user agent string observed in a household. Naturally, people use several browsers to explore the web, and several persons are aggregated in a household. Privacy requirements limit any different granularity. In chapters 4 and 5 I limit the data to i) the anonymized client IP address, ii) the name of the contacted server, and iii) the timestamp of the TCP connection. I further anonymized the server name in the datasets that I contribute to the community which contains data collected from my campus.

For a broader discussion on data collection and ethics for the different stakeholders in the Internet scenario, see appendix A.

1.4 Final notes for the reader

Part of the work presented in this thesis was developed in collaboration with other researchers. Each chapter reports the references to our published work (in some cases, papers under review). I was responsible for most of the work that I am reporting in all the chapters: therefore, I can state that the thesis is primarily based upon my personal work and hence I am using the first person singular pronoun throughout the manuscript.

The few sections that I wrote where I was not the original conceiver of the ideas and methodologies are clearly highlighted. Here, I report them:
• The methodology to classify Core and Support domains introduced in section 4.3 was primarily designed by my colleague Ing. Martino Trevisan.

• In section 5.3, I summarize TribeFlow [8], a methodology to model each user as a random surfer over latent environments, primarily designed by Prof. Flavio Figueiredo, with whom I am collaborating.

• The proofs presented in sections 6.3.4, 6.4 and 6.5 were primarily work of my collaborators Prof. Emilio Leonardi, Michele Garetto and Carla Chiasserini.
Chapter 2

Detecting user-actions from HTTP traces

Most of the work I am presenting in this chapter is taken from my published paper [9].

2.1 Introduction and terminology

A user-action is the action of requesting a URL by a user to fetch a web-page, triggered by an interaction with a browser. The request can be done either by clicking on a hyperlink, by typing the URL in the browser address bar, or through bookmarks. Knowing the URLs visited by users is fundamental for many applications. For instance, network security and forensics applications [10] profit from logs of HTTP requests to detect when and how users get infected by malware or viruses. Equally, URLs requested by users can unveil web content popularity [11], and be used for ranking and promoting content [1]. One possible way of obtaining user-actions is by passively observing network traffic [12, 13]. One can extract HTTP requests and save requested URLs. Similar information can be obtained from proxy or firewall logs too. Many off-the-shelf flow meters are able to export HTTP traces along with traditional flow-level information [14], thus making HTTP traces sometimes readily available for processing.

I am interested in studying how people surf the web, based on the analysis of raw HTTP logs. The fundamental technical challenge is to extract user-actions from
Figure 2.1 Example of a single web-page structure. In this case the main HTML file (orange object) is the URL request of the user-action and the others (yellow objects) are embedded objects, i.e., automatic-actions.

such noisy logs. Indeed, rendering a web-page is a rather complex process [15] that requires the browser to download HTML files, JavaScript, multimedia objects and dynamically generated content. All these objects are retrieved by the browser by means of independent HTTP requests (see Figure 2.1). Furthermore, non-interactive web applications (e.g., cloud storage clients, OS update agents, etc.) rely on HTTP to exchange data too, and all those requests are logged together with users’ activity. I define the initial HTTP request triggered by a user interaction with the browser as a user-action, and as automatic-actions all the remaining HTTP requests fired to render web-pages.

Figure 2.2 illustrates these definitions. It depicts the timeline of a user surfing the web. The user visits five web-pages, whose corresponding user-actions are marked by tall red arrows. Following each user-action, the browser fires automatic-actions to fetch objects, which is marked with short blue arrows.

User-actions thus correspond to web-pages explicitly visited by a user (e.g., http://www.example.com/index.html), and the two terms will be interchangeable in this work. For each web-page, its second-level domain (i.e., example) and top-level domain (i.e., com) are known, and combination (i.e., example.com) I will simply call a domain with a little abuse of notation. Again, refer to table 1.1 for summarization of terminology.
2.1 Introduction and terminology

I aim at characterizing browsing habits and check how they are evolving in chapter 3. I call the clickstream the list of user-actions. The clickstream is typically modeled as a directed graph, where web-pages constitute the vertices, and edges represent the movement of a user through web-pages, i.e., when the user clicks on a hyperlink. The right-hand side of figure 2.2 illustrates the clickstream extracted from the navigation example. Two components are present: user-actions 2–3–4 are reached following hyperlinks starting from user-action 1, while user-action 5 is a web-page reached independently.

I assume the availability of a monitoring infrastructure that exposes HTTP logs. Examples of such infrastructure are web proxies and network probes that extract information from packets crossing a measurement point. HTTP logs contain records about HTTP requests. Each record includes (i) the time of the request; (ii) the requested URL; (iii) the HTTP referer, i.e., the HTTP header that identifies the URL of the web page that linked to the resource being requested; and (iv) the user agent, i.e., an identification of the browser application sending the HTTP requests.

HTTP does not specify any mechanism for signaling if a request is a user-action or not. As such, HTTP URLs are indistinctly mixed in HTTP logs. Thus, the first problem I target is the identification of user-actions in the stream of HTTP requests. URLs identified as user-actions become the clickstream vertices. If a referer is present, it represents a directed edge from the URL in the referer to the user-action URL.

In this chapter I study whether machine learning algorithms can successfully be used to detect user-actions on HTTP traces. I use data from volunteers which are uniquely identified by client IP addresses and user agent strings. The deployment of any approach to identify user-actions requires the isolation of traffic per user. IP addresses and user agents may be insufficient if many users are aggregated in
Detecting user-actions from HTTP traces

NATs. Discussing alternatives to isolate traffic is out of the scope of this work. An approach based on machine learning would have many advantages. It would allow automatic tuning of parameters, could automatically learn which features are the best candidates for solving the problem, and could easily adapt to the evolution of web-pages, and to different scenarios.

I collect browsing histories of 10 volunteers for several months, while also recording all HTTP requests of their web navigation. I then extract a large number of features, which are used in experiments to select the most informative ones for the problem. I discuss results of using different datasets for training and testing and, finally, explore the performance of classification algorithms when trained with datasets of increasing sizes, by varying the number of users for training and validation.

Results show that my approach generalizes ad-hoc designed heuristics [15, 12, 1], automatically learning the patterns that characterize explicit visits (i.e., user-actions), with detection precision and recall over 90%. Moreover, I show that models built with machine learning algorithms are robust, presenting consistent performance in different scenarios.

Aiming to foster further researches and re-validations of my results, I make the datasets available (in Weka format) at http://bigdata.polito.it/clickstream.

In chapter 3 I will apply the classification algorithm described here to a long-term dataset of HTTP logs collected in an operational network.

2.2 Related work on detecting user-actions

The user-action detection problem I address is similar to web-page view identification, a part of the data cleaning process in web usage mining [16]. Web usage mining is historically the task of extracting knowledge from HTTP server-side log files. As such, this task was traditionally tailored on a per web-site basis. For instance, web-page view identification from web server logs leverages the a priori known structure of the web-site, and is performed often by discarding a manually constructed list of embedded object extensions from the logs [17, 18]. This approach does not work in my setting because I am aggregating logs from a variety of heterogeneous web servers with different web-site structures, naming conventions, Ad networks and
CDN providers. It is thus not feasible to manually construct a list of extensions to discard.

Previous works have introduced different methods for identifying user-actions from network-based HTTP traces. StreamStructure [15] exploits the referer field in HTTP requests and the widespread deployment of the Google Analytics beacon to reconstruct web-page structures from HTTP traces and identify user-actions. The authors of [12] follow a similar approach, exploiting the referer field to group requests into HTTP streams. A series of manual rules are used to decide whether a request is the consequence of a user-action or not. Precision and recall above 90% are claimed on synthetic traces. Finally, the authors of [1] present a heuristic to identify user-actions that operates only with the HTTP requests. The proposed heuristic is shown to scale well in high-speed networks, claiming 66%–80% precision and 91%–97% recall, depending on parameter choices.

Previous works however present some drawbacks. First, they are all based on manually tuned heuristics. Whereas the heuristics are shown to produce good results, they require time-consuming work to be configured, and the procedure might even need to be performed periodically to adapt parameters as web-pages evolve. Second, given the difficult to obtain HTTP traces simultaneously to ground truth of actual users’ requests, previous works have mostly validated proposals using limited synthetic datasets. In fact, ground truth datasets are generally built by automatically visiting arbitrary links, which may miss (create) artefacts (not) seen in real traces. More important, synthetic datasets do not contain the variety of browsers and behaviours of real traces. This is worrisome, given that some browsers even skip filling the referer fields in some cases [13], thus potentially affecting heuristics’ precision when deployed in practice.

In contrast to previous efforts, my approach is fully automatic, does not require manually tuned parameters and is validated with traces from real users.

2.3 Annotated datasets of volunteers’ activity

Training a classifier and testing its performance require data in which the ground truth is known, i.e., requests are annotated with class labels. In the studied scenario, HTTP logs in which requests are annotated as user- or automatic-actions are needed.
Instead of only building such traces in a controlled environment as done in previous works, I rely on real traces from actual end-users to build the ground truth.

To obtain the ground truth of user-actions, I rely on volunteers. Specifically, I collect the visited web-pages from volunteers’ PCs by extracting their browsing history from three major browsers: Safari, Chrome and Firefox. Referring to figure 2.2, a volunteer’s browsing history exposes the timeline of user-actions. It includes (i) timestamps of web-page visits; (ii) the requested URLs; and (iii) codes describing web-page transitions – e.g., whether the visit resulted in a redirection to another web-page. In total, I observed more than 12 000 visits to more than 2 000 web-sites in 3 months of browsing activity of 10 volunteers.

During the same period, my campus network was instrumented to passively collect the raw HTTP logs of these volunteers by observing the traffic flowing in and out of my university. I use Tstat (see section 1.2 for details) to perform the collection. This tool exposes information from both the TCP and HTTP headers, including (i) TCP flow-ID, (ii) timestamps; (iii) requested URLs; (iv) user agent; (v) referer; (vi) content type; (vii) content length; and (viii) status code. Referring to figure 2.2, Tstat exposes the timeline of all HTTP requests observed in the network. All in all, I record more than 0.6 million HTTP requests related to the volunteers.

I next match entries in HTTP logs with entries extracted from browsing histories to label user-actions. The matching of entries however requires care. I primarily use the URL and timestamps as keys, but web-page redirections may create issues. For instance, shortened URLs or web-page redirections are logged in different ways by various browsers. I decide to label as user-action the last request in a redirection chain that is present in both HTTP logs and browsing histories. Finally, I filter out any HTTP requests coming from web browsers I did not capture data (e.g., Internet Explorer), since they produce user-actions that I am not able to label. I however leave requests of background applications, since by definition they are automatic, such as Dropbox or Windows Update, and my goal is to train models able to discern
such requests. At the end of this process, I mark about 2% of all HTTP requests as actual user-actions.

Figure 2.3 summarizes the data collection and preparation methodology.

2.4 User-action classifier design

This classification problem consists in identifying a visited URL as either user-action or automatic-action. As I have shown in section 2.2, in the past this problem has been faced by designing ad-hoc heuristics driven by domain knowledge, e.g., by rebuilding the web-page structure [15, 12], or manually building blacklists and simple tests [1]. I here revisit the problem and introduce a machine learning methodology. It is given a labeled dataset where the classes of observations are known. Observations are characterized by features, i.e., explanatory variables that describe observations. The classifier uses the knowledge of the class to build a model that, from features, allows it to separate objects into classes. The approach is generic, and several algorithms are available to select the most important features, build the model and then make classification decisions.

2.4.1 Feature extraction

Features play a key role in classification. Instead of a priori selecting features that I believe might be useful for classification, I follow the best practice of machine learning and extract a large number of possibly generic features. I let the classifier build the model and automatically choose the most valuable features for the goal.

Table 2.1 lists the features extracted from traffic traces. I consider 17 features that can be coarsely grouped into four non-independent categories: (i) based on referring relations among URLs; (ii) based on timestamps; (iii) describing properties of objects; and (iv) describing properties of URLs. Some features are boolean or categorical, i.e., they can take a limited number of labels, while others are counters or real-valued. Features are sorted by their Information Gain (IG) with respect to the user-action class in the dataset of volunteers’ activity, a notion that I will discuss later. Previous works that use some of these features as part of the manually tuned heuristics are reported in the table.
Detecting user-actions from HTTP traces

Table 2.1 Features and IG with respect to the user-actions.

<table>
<thead>
<tr>
<th>Feature</th>
<th>referer</th>
<th>Time</th>
<th>Object</th>
<th>URL</th>
<th>Type</th>
<th>IG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Children [1][12][15]</td>
<td>x</td>
<td></td>
<td></td>
<td>Count</td>
<td></td>
<td>0.2706</td>
</tr>
<tr>
<td>Content Type [1][12][15]</td>
<td></td>
<td>x</td>
<td></td>
<td>Cat.</td>
<td></td>
<td>0.0287</td>
</tr>
<tr>
<td>$\Delta_t$ – Previous Request</td>
<td>x</td>
<td></td>
<td></td>
<td>Real</td>
<td></td>
<td>0.0140</td>
</tr>
<tr>
<td>HTTP Status Code [12]</td>
<td></td>
<td></td>
<td>x</td>
<td>Cat.</td>
<td></td>
<td>0.0061</td>
</tr>
<tr>
<td>URL length</td>
<td></td>
<td></td>
<td></td>
<td>Count</td>
<td></td>
<td>0.0060</td>
</tr>
<tr>
<td>$\Delta_t$ – Sibling</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Real</td>
<td></td>
<td>0.0048</td>
</tr>
<tr>
<td>URL in ad List</td>
<td></td>
<td></td>
<td>x</td>
<td>Bool</td>
<td></td>
<td>0.0040</td>
</tr>
<tr>
<td>$\Delta_t$ – Parent [12][15]</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Real</td>
<td></td>
<td>0.0036</td>
</tr>
<tr>
<td>Content Length [12]</td>
<td></td>
<td></td>
<td>x</td>
<td>Count</td>
<td></td>
<td>0.0027</td>
</tr>
<tr>
<td>Parent Status Code</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Cat.</td>
<td></td>
<td>0.0016</td>
</tr>
<tr>
<td>Has referer?</td>
<td>x</td>
<td></td>
<td></td>
<td>Bool</td>
<td></td>
<td>0.0014</td>
</tr>
<tr>
<td>Max $\Delta_t$ – Child</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Real</td>
<td></td>
<td>0.0010</td>
</tr>
<tr>
<td>Parent Content Type</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Cat.</td>
<td></td>
<td>0.0007</td>
</tr>
<tr>
<td>Referer in ad list</td>
<td>x</td>
<td></td>
<td>x</td>
<td>Bool</td>
<td></td>
<td>0.0005</td>
</tr>
<tr>
<td>Max Length – Child</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Count</td>
<td></td>
<td>0.0005</td>
</tr>
<tr>
<td>Min $\Delta_t$ – Child</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Real</td>
<td></td>
<td>0.0003</td>
</tr>
<tr>
<td>Parent Content Length</td>
<td>x</td>
<td></td>
<td>x</td>
<td>Count</td>
<td></td>
<td>0.0002</td>
</tr>
</tbody>
</table>

Features are extracted from HTTP logs. Given a URL, I calculate the time interval ($\Delta_t$) from the previous request from the same browser. If the request has a referer, I call the URL in the referer its parent. I also determine whether a URL in a request has children, i.e., subsequent requests that have this particular URL in the referer field. Based on parent-child relations, I extract the number of children, the time interval between the request and its eventual parent, and the time interval between the request and its last sibling, i.e., previous request sharing the same parent. If the request has children, I compute the minimum and maximum time to see a child.

I consider features in server responses, such as the Status Code, Content Type and Content Length. I also augment the feature set with statistics of the request of the parent (if it exists), e.g., the Content Length, Content Type and Status Code of the parent request. Finally, I include features that describe the URL strings. I count the number of characters in the URL and I check if the URL (or the referer) is included in a blacklist of terms associated with advertisement domains.
2.4.2 Classifier choice

Given the heterogeneity of features and their diverse nature, the choice of which classification model to adopt requires ingenuity. For instance, algorithms based on the notion of distance such as support vector machines or nearest neighbor methods are particularly sensitive to the presence of boolean and categorical features. Similarly, the presence of dependencies between the features challenges regression based classifiers. Generally, when there are complex dependencies among features, decision trees and neural networks offer the best performance [19]. Here, I am not interested in providing a complete assessment of which classifier performs the best, but rather to coarsely observe if there are significant differences for this specific case. As such, I consider:

**Decision Tree (DT):** it is a tree-like classifier for making sequential decisions on features [20]. Internal nodes represent tests, branches are the outcomes of tests, and leaves represent classes. I use J48 – an open source implementation of the C4.5 decision tree training model.

**Random Forest (RF) [21]:** it improves and generalizes decision trees. It constructs a multitude of decision trees at training time using subsets of features, outputting the class that is the mode among those trees. RF is more robust to over-fitting than DT.

**Bayesian Network (BN):** BN [22] is a probabilistic graphical model that represents a set of features and their conditional dependencies via a Directed Acyclic Graph (DAG). Such networks are factored representations of probability distributions that generalize the naive Bayesian Classifier.

**Multi-layer perceptron Neural Network (NN):** it is a feedforward neural network that maps features into classes [23]. It consists of multiple layers of nodes in a directed graph, where each node is a processing element with a nonlinear activation function. Training is performed with the backpropagation algorithm.

I use the implementations offered by Weka in my experiments.¹

2.4.3 Performance metrics

The classification performance measures the ability to correctly return the class of an object. Performance is typically summarized using accuracy, i.e., the fraction of objects from any class that are correctly classified. Accuracy is often misleading, especially when object classes are unbalanced, i.e., a naive classifier returning always the most popular class would achieve a high accuracy. In such cases, per-class performance metrics must be considered.

For a given class \( c \), we may have a True Positive – \( TP(c) \) – when the returned class \( c \) is correct; a False Positive – \( FP(c) \) – when incorrectly returned as \( c \); a False Negative – \( FN(c) \) – when incorrectly non identified as \( c \); and True Negative – \( TN(c) \) – when returned correctly as not \( c \).

Therefore, given I am interested in user-action classification, I also evaluate performance metrics related to this class:

(i) Precision: the fraction of requests correctly classified as user-action (the number of true positives among the requests that the classifier selected as user-actions)

\[
\text{Precision}(c) = \frac{TP(c)}{TP(c) + FP(c)};
\]

(ii) Recall: the fraction of user-actions that the classifier captures (number of detected user-actions over the total number of user-actions)

\[
\text{Recall}(c) = \frac{TP(c)}{TP(c) + FN(c)};
\]

(iii) \( F - \text{measure} \): the harmonic mean of Precision and Recall for the class

\[
F - \text{measure}(c) = \frac{2 \cdot \text{Precision}(c) \cdot \text{Recall}(c)}{\text{Precision}(c) + \text{Recall}(c)}.
\]

After the model has been built on a dataset, I want to estimate its performance for selecting the best classification setup, i.e., the best tuning parameters. I use the standard stratified 10-fold cross-validation. The training dataset is partitioned into 10 sub-samples of equal size. Of the 10 sub-samples, one is retained for measuring the performance, and the remaining 9 sub-samples are used for training the model. This process is repeated 10 times, with each of the 10 sub-samples used exactly once as the validation data. Finally, the 10 results are averaged to produce a single estimation.
2.5 User-action classifier performance

In this section I provide a performance evaluation of the user-action classifier on ground truth traces.

2.5.1 Feature relevance

The central idea when doing feature selection is that the data may contain irrelevant or redundant features. To check which features are relevant to separate user- from automatic- actions, I compute the Information Gain, also known as the mutual information. It quantifies the reduction in entropy caused by partitioning the dataset according to the values of the specific feature. In table 2.1, I rank features: the higher the information gain, the higher is the information about the user-action class that the specific feature carries in isolation. Results give an initial overview of the discriminating power of each feature in isolation. I will discuss later the performance of classifiers trained with subsets of features.

We see that the Number of Children is by far the feature with the highest IG. Content Type is well-ranked as well. These results confirm the intuition of prior work [1, 12, 15] that referer relations and the analysis of Content Types strongly help in user-actions detection. Next to these features, I find the time interval ($\Delta t$) between consecutive requests of a single browser, and some other features that are independent of the referer, such as HTTP Status Code and Size of URL. Note that these features independent of the referer may help in solving artefacts related to lack of referer in HTTP requests [13]. To this ends, classifiers that do not use referer are worth to investigate. I will train classifiers that use only non referer based features in the next section.

2.5.2 Performance of classifiers and feature sets

I evaluate the performance of the different classification models and different feature sets. Table 2.2 reports the classification accuracy, and F-measure, precision and recall for the user-action class in the 10-fold cross validation experiments. These results are obtained using all features listed in table 2.1.
Table 2.2 Performance of the classifiers. F-measure, precision and recall for the user-action class. Cross-validation with PC volunteers’ training set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>F-measure</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>0.996</td>
<td>0.906</td>
<td>0.905</td>
<td>0.907</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.996</td>
<td>0.912</td>
<td>0.891</td>
<td>0.943</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>0.989</td>
<td>0.774</td>
<td>0.723</td>
<td>0.835</td>
</tr>
<tr>
<td>Neural Network</td>
<td>0.994</td>
<td>0.888</td>
<td>0.860</td>
<td>0.917</td>
</tr>
</tbody>
</table>

Table 2.3 F-measure with different features sets with volunteers’ dataset. Cross-validation results shown.

<table>
<thead>
<tr>
<th>Model</th>
<th>All Features</th>
<th>Top-5 Features</th>
<th>No referer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>0.906</td>
<td>0.872</td>
<td>0.846</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.912</td>
<td>0.866</td>
<td>0.831</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>0.774</td>
<td>0.819</td>
<td>0.700</td>
</tr>
<tr>
<td>Neural Network</td>
<td>0.822</td>
<td>0.752</td>
<td>0.529</td>
</tr>
</tbody>
</table>

We see in table 2.2 that the accuracy is higher than 99% for the four classification models. Recalling that we have about 98% of automatic-actions, i.e., classes are strongly unbalanced, a naive classifier that always returns “automatic” would have about 98% accuracy. As such, I focus on performance for the user-action class. All three machine learning classifiers deliver very good performance. Neural network has the lowest F-Measure among the three. The J48 decision tree F-Measure equals to 90.6%, while the Random Forest performs marginally better.

Given a decision tree is simpler than a random forest, but with similar performance, I decide to use the former from now on. An appealing characteristic of decision trees is the easy interpretation of the built model. Manually inspecting the tree, we see that indeed features with high IG (e.g., number of children) are among the top features of the tree. The classifier code can be downloaded from http://bigdata.polito.it/clickstream.

Table 2.3 reports the F-measure for the user-action class considering different feature sets. Experiments with the best-ranked features (Top-5 Features) result in minor performance variations. Focusing on the second column of results in table 2.3, notice how numbers for the classification using only the top-5 features (see table 2.1 for the list) are very similar to the ones with all features. The F-Measure
for the Random Forest is reduced from 91.2% to 86.6%. Then the mix of features representing inter-timing of requests, referer, as well as properties of responses and URLs provides very good classification, despite the complexity of the problem.

The last column in table 2.3 shows how the performance varies when all features in table 2.1 that have relation to referer are excluded. We see that the performance decreases when compared to previous experiments. However, even without attributes related to the referer, the J48 decision tree reaches a precision of 89% and recall of 80% – i.e., F-Measure of 84.6%. Bayesian Networks and Neural Networks classifiers show lower-quality results.

### 2.5.3 Testing on smartphone traffic

Training and testing have been done so far considering annotated traces from browsers running on PCs. It is not clear whether the classifier would perform well for other browsers. In particular, I am interested in validating the methodology for mobile browsers, i.e., apps that can be used to navigate through multiple web-pages and domains. Nowadays these browsers are extremely popular and present different characteristics from PC browsers. To answer this question, I collect traces by manually visiting popular HTTP web-sites, taken from top 100 Alexa web-sites\(^2\) and by randomly following two links inside each of them. I performed the experiment by re-opening the URLs with different apps in an Android smartphone, while connected via WiFi to the campus network instrumented with Tstat. I chose four different browsers: Chrome, UC browser, Instagram and Facebook. Instagram and Facebook are apps primarily thought to exploit their internal services, but that allow as well following links to external web-pages in an in-app browser. The traces have been collected by visiting these web-pages manually, waiting for each web-page to be fully-loaded before visiting the next web-page. Even if this behaviour cannot be considered totally natural as for the PC volunteers, the dataset has some ingredients of real user interactions. Moreover, I do not filter out background traffic of the smartphone.

At the instrumented network, \(\approx 5000 - 8000\) HTTP requests have been recorded, depending on the browser, which I classify using the previously described decision tree. The results are in table 2.4. Performances are inline with previous experiments,\(^2\) [https://www.alexa.com/topsites](https://www.alexa.com/topsites), accessed January 2018.
Table 2.4 Performance of the classifiers on smartphone applications. F-measure, precision and recall for the user-action class. Decision tree tested on smartphone traces for different apps.

<table>
<thead>
<tr>
<th>Testing Dataset</th>
<th>Accuracy</th>
<th>F-measure</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>0.999</td>
<td>0.935</td>
<td>0.935</td>
<td>0.935</td>
</tr>
<tr>
<td>UC browser</td>
<td>0.996</td>
<td>0.863</td>
<td>0.923</td>
<td>0.810</td>
</tr>
<tr>
<td>Facebook</td>
<td>0.998</td>
<td>0.931</td>
<td>0.902</td>
<td>0.961</td>
</tr>
<tr>
<td>Instagram</td>
<td>0.998</td>
<td>0.930</td>
<td>0.890</td>
<td>0.973</td>
</tr>
</tbody>
</table>

i.e., precision and F-measure close to 90%. There are very few false positives and even less false negative. Only the UC browser shows a lightly smaller value of recall with respect to the others. Indeed, this browser performs compression of the web pages and some requests are dropped. Therefore, they cannot be recognized as user-actions.

The validation that I performed on different mobile browsers suggests that my machine learning classifier, although trained on PC datasets, adapts well also to traffic towards other devices.

2.5.4 Training set size

To get more insights on the performance of the decision tree classifier, I run experiments varying the training set size. On each round, I consider an increasing number of volunteers in the training set. I then assess performance (i) using 10-fold cross validation on the same training data; (ii) validating the model on the remaining volunteers’ traces that were not included in the training set, i.e., an independent test set.

The results are on figure 2.4, where F-measure for the user-action class is depicted. Considering the cross validation estimates, the F-measure reaches a plateau when two volunteers are considered. That is, the classifier is able to model the browsing habits of the volunteers included in the training set. More importantly, the validation with independent users shows consistent results, reaching more than 90% of F-measure when seven or more volunteers are in the training set. In a nutshell, the behaviour of
2.5 User-action classifier performance

Figure 2.4 Effects of varying the number of volunteers for training the decision tree. F-measure considered.

Table 2.5 Performance of heuristic presented in [1] tested with volunteers dataset and compared to the decision tree.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>F-measure</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>0.996</td>
<td>0.906</td>
<td>0.905</td>
<td>0.907</td>
</tr>
<tr>
<td>Manual Heuristic [1]</td>
<td>0.988</td>
<td>0.784</td>
<td>0.711</td>
<td>0.870</td>
</tr>
</tbody>
</table>

the independent volunteers has been learned from other users. These figures provide additional evidence that the produced model is robust and generic.

2.5.5 Comparison with methodology by Ben-Houidi et al.

I apply the heuristic proposed by Ben-Houidi et al. [1], which was manually crafted using domain knowledge, to my datasets, to contrast its performance with the machine learning approach I propose. Results are in table 2.5, where the results of the trained decision tree are obviously the same as in table 2.2.

Numbers in the table are compatible with what is reported in [1] – i.e., precision is on the 66%–80% range, whereas recall is close to previously reported results (91%–97% range). More interesting, all machine learning alternatives of table 2.2 have much better performance than the manual heuristic thanks to my larger set of features, and the higher variability in actual user browsing habits that I used to train...
the models. Recall that the manual heuristic we compare to has been trained and tested on a smaller ground truth trace built in a controlled environment.

It confirms that the machine learning approach delivers better performance than previous work, offering thus a solid alternative for classifying user-actions.

### 2.6 Conclusions and discussion

This chapter discussed the use of machine learning classifiers for the identification of explicit user-actions in passive HTTP traces. In particular, I have (i) evaluated different subsets of features and machine learning models; (ii) studied how the performance of algorithms varies when different numbers of users are used for training; (iii) validated the performance on traffic coming from different typology of applications and devices.

Results show that my approach generalizes ad-hoc designed heuristics, automatically learning the patterns that characterize explicit visits, with detection precision and recall over 90%.

Some of my analyses have limitations. First, I acknowledge that the data of volunteers’ activity is not representative of the complete Internet population, nor it covers all types of client browsers. Yet, it is better than previous works and the successful application to smartphones traffic gives a good confidence. Second, the machine learning approach may also require periodical updates of the classification model to keep pace with the web evolution. This will require mechanisms to obtain ground truth data from users with different profiles. Finally, although I compared three distinct models, many other algorithms and configurations of parameters could be tested, as well as mechanisms to evaluate and avoid over-fitting during training.

In chapter 3 I will apply the classification algorithm here described to a long-term dataset of HTTP logs collected in an operational network.
Chapter 3

Longitudinal characterization of browsing habits

Most of the work I am presenting in this chapter is taken from my paper [24], currently under review.

3.1 Introduction

Since its introduction, the web has become a fundamental mean to access information. Technology has evolved, from simple static web-pages to dynamic applications that let users search for content, buy goods, spend time on social networks, i.e., to browse the web. Understanding how people interact with the web has been always a fascinating problem [25–28] for a variety of purposes like improving search engines [2], comparing rankings in the web [29, 30], recommending content to people [4], or increasing privacy and security [5].

Browsing activities have been typically modeled using graphs – or clickstream graphs as they are typically called – where vertices are the visited web-pages, and edges are the followed hyperlinks. They capture the paths that users take when navigating through web-sites. The right-hand side of figure 2.2 illustrates the clickstream graph extracted from the navigation example. Two components are present: user-actions 2–3–4 are reached following hyperlinks starting from user-action 1, while user-action 5 is a web-page reached independently.
The evolution of the web, obviously, changes how users interact with it. We today witness the predominance of a handful of popular online services [31] and the rise of mobile devices. How are these factors changing the way we browse the web?

In this chapter I am interested in answering the following two questions:

• How are the clickstream graphs affected by the web evolution over the past years?

• What are the differences between clickstream graphs from different browsing devices (e.g., PCs and smartphones)?

I provide a longitudinal characterization of the clickstream graphs. Fundamental to answer these questions is the availability of data. Previous studies are either outdated [32], or focused on small sets of volunteers [33–35], on user interactions with search engines [36] or proxy logs [37]. Only few studies have used passive network traces to study browsing behaviour [26, 12]. In this chapter, I leverage a three-year long anonymized dataset of passive measurements from an ISP network, thus offering a privileged observation point. The probes monitored 25 000 households and observed more than 64 billion HTTP requests. From this dataset, I extract the subset of requests related to explicitly visited web-pages (user-actions) by adopting the novel approach based on machine learning algorithm showed in chapter 2.

Given the requested web-pages and the hyperlinks followed by people, I build clickstream graphs for each browser in a household. In total, I construct 5.5 million graphs corresponding to over 1 billion visited web-pages. I then exploit this dataset to investigate browsing habits, providing the evolution over three years, and carefully characterizing differences in usage according to device types. In summary, this chapter makes the following contributions:

• I present a characterization of clickstreams that differs from previous efforts [12, 11] for (i) covering a large population during a long consecutive period of time, and (ii) accounting for different device types used to browser the web at home.

• I contribute to the community a three-year long dataset of anonymized clickstreams, covering thousands of households in Europe, accessible at http://bigdata.polito.it/clickstream. To the best of my knowledge, this is one of the
largest datasets that includes clickstream graphs from regular Internet users, browsing with multiple devices.

I focus on global patterns, highlighting when there are general trends in surfing habits, rather than unexpected but rare events. My analysis confirms and precisely quantifies many intuitions about the way people navigate the web, besides leading to a number of interesting findings such as:

• Search Engines (SEs) and Online Social Networks (OSNs) are among the preferred means to discover content. As of 2016, 54% of web domains were visited starting from Google, and 9% (6% in 2013) starting from Facebook. SEs are starting point of longer and deeper navigation, while content promoted by OSNs typically generates visits to a single or very few web-pages. Interestingly, OSNs are much more important to discover content on smartphones than on PCs, a result previously not highlighted.

• Web-page complexity has continuously increased from 2013 to 2016, with URLs intentionally visited by users going from 2% to 1.5% of the total number of URLs requested by browsers.

• The number of devices and applications used to browse the web at home has increased significantly, with smartphones and tablets accounting for 29% and 9% of the visited web-pages in 2016, respectively. Users are interacting more frequently with the web from their smartphones at home than in the past [12]. However, in a session on a mobile app on average only 5 web-pages are visited, in a time span of only 2 minutes.

• When considering the number of visited web-pages, we observe that 50% of the clickstream graphs include less than 27 web-pages per day for PCs (8 for smartphones), belonging to less than 9 domains (4 for smartphones). Considering consecutive visited web-pages, i.e., a path, we observe that people stay in very few domains, even when navigating through hundreds of web pages. These numbers have mostly remained constant over the years, despite changes in devices and applications used to browse the web.

• Encryption has gained momentum in the web with many popular domains migrating to HTTPS. We clearly see the impact of HTTPS on properties of the
clickstream graphs. Still, in June 2016, only around 13% of the domains are served (partly or totally) in HTTPS, and 85% of the encrypted traffic is related to the top 20 content providers, like Google and Facebook.

These findings and the contributed dataset have several implications to the Internet actors. For example, they can (i) help advertisers to make informed decisions on whether to target ads campaigns on mobile or PC users; (ii) help network operators to understand interests of users and recommend products and services; (iii) help researchers to investigate privacy aspects related to properties of clickstreams learned from traffic; and, more generally, (iv) help the research community to study the place of web technologies in people’s life.

Please refer to section 1.1 for more details on the problem and on the terminology used in this chapter.

3.1.1 Scope and limitations

The scope of my study is obviously limited by the coverage and characteristics of the dataset.

First, the evaluated dataset is limited to the non-encrypted part of the web. It however covers a particularly interesting period, in which the usage of HTTPS has grown from negligible to noticeable percentages. Despite the growth on HTTPS usage, the majority of the domains were still not encrypted by the end of the data capture in 2016. Moreover, transitions from popular encrypted domains to the unencrypted ones are still visible in the analysis. This happens because early adopters of full HTTPS deployments are large content promoters (e.g., Google and Facebook) that still inform non-encrypted domains the origin of visits. As such, we have no information about actions performed inside these encrypted domains, but we can see the transition when users eventually leave them towards unencrypted domains. An encrypted domain appears as a single vertex in a clickstream graph, connected to all vertices representing domains visited from it.

Second, no information to identify people exists in the dataset. Section 3.3 will show that households (i.e., home Internet installations) are identified by anonymized keys, and browsers by user agent strings. Analyses are thus performed in a per-browser level – i.e., each user agent string observed in a household. Naturally,
3.2 Related work on user behaviour and clickstream analyses

people use several browsers to explore the web, and several persons are aggregated in a household. Privacy requirements however limit any different granularity.

Third, the evaluated dataset includes only a regional sample of households in Europe. Users in other regions may have diverse browsing habits that result in different clickstreams. Equally, mobile devices have been monitored only while connected to home WiFi networks. As such, my quantification of browsing on mobile terminals is actually a lower-bound, since visits while connected to other technologies are not captured.

Last, as in any large-scale analysis of real-world measurements, many preparation steps have proven essential to clean up spurious data and reduce biases on results. For instance, I have observed non-standard implementations of HTTP protocols by some browsers that prevent the reconstruction of the clickstreams in certain situations. Equally, I was faced with many challenges to reconstruct clickstreams on mobile terminals, given the diverse ways that mobile apps operate.

I will elaborate further about these technical aspects in section 3.3.

3.2 Related work on user behaviour and clickstream analyses

In the last decade, several works focus on the behaviour of end-users – e.g., to determine how often web-pages are revisited and how users discover and arrive to web-pages. The authors of [35, 38] characterize web usage exploiting few volunteers’ browsing histories. They find that navigation based on search engines and multi-tabbing are changing the way users interact with browsers – e.g., direct web page visits based on bookmarks and backtracking are becoming less popular. A similar study is presented in [34], based on device instrumentation and a small user population, and in [37], based on proxy logs. The closest to my work is [11], which leveraged the Yahoo toolbar to analyze web-page views from a large population of heterogeneous users over a period of one week. Some of my findings confirm theirs (e.g., deep browsing after leaving search engines) while others present different figures compared to what they discovered (e.g., I observe a higher weight of social networks in referral share and less no-referrer traffic). However, with the very fast
evolution of the networking technologies and the web, the question whether these results still hold nowadays is raised. My study answers it on various aspects.

With the emergence of connected mobile devices, the works focusing on mobile users’ behaviour have multiplied. The authors of [39] propose a taxonomy of usage of the Internet for mobile users, where data are extracted thanks to contextual inquiries with volunteers. They retrieve three already known categories (information seeking, communication and transaction) and identify a new one: personal space extension, anticipating the wide usage of cloud storage systems. The authors of [40] focus on the usage of mobile applications, showing that sessions are generally very short (less than one minute on average). The authors of [41] perform a survey on mobile Internet behaviour, concluding that the approach to measure the usage (passive/active, objective/subjective, etc.) could heavily impact the results. A recent work [42] analyses the Wi-Fi access logs of a city shopping mall, showing that the user revisit periodically the same web content. Considering HTTPS usage on mobile devices, the authors of [43] study HTTP/HTTPS deployment in mobile networks, finding that HTTPS is mainly used by large Internet services. My work confirms many of these trends. However, I not only present aggregated statistics about protocol usage, but also extract clickstreams from HTTP traffic.

The comparison between mobile and PC Internet usage gave rise to a lot of studies. For example, the authors of [44] present a comparison of objects retrieved from PCs and smartphones, and implications for caching, but without distinguishing user-actions. The authors of [33] compare smartphone and PC navigation, concluding that web-page revisits are rare in smartphones, while bookmarks are more widely used on smartphones than on PCs. The authors of [45] study the differences in searching behaviour of mobile, tablet and PC users. They show that most clicked web-sites depends on the used device, suggesting that these differences should be taken into account to design specific ranking systems.

On a more sociological side, the authors of [46] note that the computers can increase the use of capital enhancing activities and that to palliate the unavailability of Internet access using mobile devices is not enough to restrain the digital divide. The authors of [47] note that mobile usage induces a checking habit for smartphone, consisting of quick and repetitive content inspection.

Considering clickstream analyses, the work presented in [32] studies the number of web-pages that a user visits within a single web-site, while the one in [26] analyses
the relations between queries on a search engine and followed paths. On another side, the authors of [48] define navigation profiles considering data exported by a browser toolbar in Russia, showing that the navigation path leading users to web pages characterizes properties of the destination web-page. Finally, the authors of [49] focus primarily on the categories of web-sites that mobile users access.

In contrast to my work, all these previous works rely on somehow limited vantage points. Here I capture HTTP traces from a large ISP to extract clickstreams, considering different devices and users while connected at residential networks. Moreover, thanks to the three years of data capture, I provide a comprehensive analysis on how browsing behaviour is evolving over time. Equally importantly, I study more aspects like the graph structure of clickstreams and the comparison between OSNs and SEs.

The authors of [12] present analyses of clickstreams from passive traces collected for two months. My work reappraisals some results in [12] – e.g., I confirm that clickstreams are small and restricted to few domains. However, the duration and richness of my data shed light on novel aspects of browsing habits. For instance, in contrast to the previous work, I show that browsing from mobile devices is getting more and more frequent, suggesting that it will soon surpass PC browsing even for users connected at home networks.

The web graph can be obtained thanks to active crawlers [27, 28, 2]. My work is orthogonal to those efforts, since they miss how users interact with the web. Our work can be of interest for the authors of [50], who introduce a system to cluster clickstreams, aiming at mining knowledge from them. This dataset is a unique source for such analyses, since it covers a large user population for a long period.

### 3.3 Datasets: ISP traces

I evaluate a long-term dataset of HTTP logs captured in a European ISP network to study how users interact with the web and how such interactions evolve over time. Three probes running Tstat (see section 1.2) have been installed in Points of Presence (PoPs) in different cities in the same country, where they observe about 25 000 households overall. Each household is assigned and uniquely identified by a static IP address. Users connect to the Internet via DSL or FTTH, using a single access
gateway offering Ethernet and WiFi home network. The sets of households may have slowly changed over 3 years. However, I always consider daily and monthly statistics which are marginally affected by such changes. In a household several users using different applications and devices may connect: I define the browser clickstream as the user-actions performed by a particular browser in a household, and thus characterized by the pair of anonymized IP address and user agent string. I use the term browser to refer to any specific application that uses HTTP to fetch resources from the web, e.g., traditional web browsers, mobile apps, etc. Obviously, a browser clickstream is not equivalent to the entire activity of a physical person since the same person could use different browsers.

Tstat was used to capture HTTP logs, and it saves flow-level statistics simultaneously to the collection of HTTP logs. These statistics include the number of exchanged bytes/packets, flow start/end timestamps, and, for each HTTP request/response pair, timestamp, server hostname, client IP address, URL, referer, user agent string, content type, content length, and status code. To reduce the privacy risks, Tstat anonymizes IP addresses and removes parameters from URLs in both GET and in the referer fields. I rely on the Universal Device Detection library\(^1\) to parse user agent strings and infer the type of devices (e.g., PC, tablet, smartphone, etc.) and the application used. The library operates by matching the user agent strings against a collection of regular expressions describing the different devices.

Tstat also implements DPI mechanisms to identify application layer protocols, such as HTTP and HTTPS. Moreover, Tstat records the server Fully Qualified Domain Name (FQDN) the client resolved via previous DNS queries, using its DNS-Hunter plugin [7]. This mechanism allows us to know which FQDN the client contacted when accessing a given server IP address, and track the usage of HTTP/HTTPS per domain.

I evaluate data collected during 3 years from July 2013 until June 2016. Table 3.1 summarizes the datasets used in this chapter. In total, Tstat logged information about more than 64 billions of HTTP requests, from which 1.1 billion user-actions are identified. Note that the probes have had some outages during the course of the data collection – the exact number of days in which each probe was active is shown in table 3.1. I will not show results for the analysis affected by partial outages.

Table 3.1 Summary of the ISP traces.

<table>
<thead>
<tr>
<th>Name</th>
<th>Households</th>
<th>HTTP Requests</th>
<th>User-actions</th>
<th>Active days</th>
</tr>
</thead>
<tbody>
<tr>
<td>PoP 1</td>
<td>≈ 10000</td>
<td>28.8 billions</td>
<td>477 millions</td>
<td>1068</td>
</tr>
<tr>
<td>PoP 2</td>
<td>≈ 13000</td>
<td>30.3 billions</td>
<td>494 millions</td>
<td>752</td>
</tr>
<tr>
<td>PoP 3</td>
<td>≈ 2000</td>
<td>5.3 billions</td>
<td>79 millions</td>
<td>600</td>
</tr>
<tr>
<td>Total</td>
<td>≈ 25000</td>
<td>64.4 billions</td>
<td>1.1 billions</td>
<td>–</td>
</tr>
</tbody>
</table>

The dataset captures how users of this ISP interact with the web. Users in different parts of the world will certainly access other domains and services. Thus, some of the results I will present next, such as about top domains promoting content, are certainly specific to this dataset. However, it covers tens of thousands of households and appear representative of the monitored country. For instance, no significant differences are observed among probes.

### 3.4 Traffic characteristics and impacts on clickstreams detection

In this section I provide an overview of characteristics of the dataset and discuss possible limitations, issues and steps I adopt to avoid biases in the analyses.

#### 3.4.1 Impact of HTTPS

My study is limited to the non-encrypted part of the web. A recent work [51] reports that HTTPS was responsible for around 45% of the user-actions by the last month of the data capture. Authors rely on direct instrumentation of Chrome and Firefox. Usage of HTTPS is similar across terminals, but with lower figures on smartphones than on PCs. For example, 38% (47%) of the user-actions are over HTTPS for Chrome on Android (Windows) on 4th Jun 2016. A steady increasing trend on the deployment of HTTPS is observed, but numbers only cover the last year of the ISP traces.

I rely on flow-level statistics saved by Tstat to gauge the effect of HTTPS during the complete duration of the evaluated traces. I quantify how many domains were
running over HTTPS and their traffic characteristics (in terms of downloaded bytes). Naturally, we cannot see the exact number of user-actions on HTTPS, but only conjecture how trends reported by [51] have evolved during the data capture.

Figure 3.1 shows the share of domains in PoP 1 relying on HTTP only, i.e., completely without encryption. The figure also reports the share of bytes on HTTP. Other probes are omitted since they lead to similar results.

The vast majority of domains relied only on HTTP in 2013 (left-most point in the figure). About 96% of the domains were exclusively running over HTTP, with another 3% using both HTTP and HTTPS. Yet, the overall number of domains using exclusively HTTP remains high – e.g., 87% of the domains, with 6% relying on both protocols in 2016. In terms of downloaded bytes, HTTP was responsible for more than 90% of the traffic in 2013, and the percentage of was reduced to around 55% in 2016 – i.e., 45% of the download traffic was encrypted by the end of the capture.

This discrepancy between domains on HTTP and the HTTP traffic, both in terms of bytes seen in the traces and user-actions reported by [51], can be explained by the fact that early-adopters of HTTPS-only deployments are among the most popular domains in the Internet. In fact, observe in the figure the increasing trend on the deployment of HTTPS in 2014, which is related to the migration of YouTube to HTTPS.
3.4 Traffic characteristics and impacts on clickstreams detection

To further understand which user-actions are missing due to HTTPS, I have studied the popular domains in July 2013 searching for cases where a significant reduction on user-actions is observed. For domains presenting a sharp decrease on popularity, I have manually investigated whether they have switched to HTTPS or not. Figure 3.2 reports the timeline of some of the relevant migrations. The traces capture the final periods of Facebook and Google Search migration to HTTPS by default. Other popular domains, such as those from Yahoo and LinkedIn, migrated during 2014. CloudFlare started offering universal HTTPS support for its customers towards the end of 2014. Wikipedia switched to HTTPS in 2015. The migration trend accelerated in the last months of the capture with several (less-popular) domains switching to HTTPS. By the end of the capture, 24 out of the top 100 domains in 2013 have switched to HTTPS.

Further checking the HTTPS traffic, I found that around 85% of the HTTPS bytes in 2016 come from the top 20 domains. Google and Facebook alone account for around 65% of the HTTPS traffic. Relevant for the analysis, a large number of HTTPS domains still pass on the referer information when users transition from the HTTPS domains to any other HTTP domain. This happens because content promoters (e.g., Google, Facebook, Yahoo, Twitter etc) have interest in informing others the origin of visits. Technical details are discussed in the next paragraph. Thus, whereas we miss user-actions inside HTTPS domains of content promoters, such as Google and Facebook, we still see the information about users’ origin when
they leave these services. For instance, Google appears as a single vertex in the graph, with edges linking it to web-pages visited after users leave its services.

**Encrypted domains and the referer**

Visits to web-pages in encrypted domains are not visible through passive measurements of the network. Moreover, RFC 7231 mandates that “a user agent MUST NOT send a referer header field in an unsecured HTTP request if the referring page was received with a secure protocol”. Therefore, referer fields on HTTPS to HTTP transitions should not be visible in the network.

However, RFC 7231 is not strictly followed by popular domains that promote content. Content promoters have incentives to pass on the referer to domains they promote, including non-HTTPS destinations. For illustration, I quote Facebook when it switches to HTTPS in 2013:

> Browsers traditionally omit the referer when navigating from HTTPS to HTTP. When you click on an external link, we’d like the destination site to know you came from Facebook while not learning your user id or other sensitive information. On most browsers, we redirect through an HTTP page to accomplish this, but on Google Chrome we can skip this redirect by using their meta referer feature.

Thus, many different strategies have been adopted over the years when moving from HTTPS to HTTP. Facebook statement explains that modern web browsers include a mechanism to allow HTTPS domains to declare whether the referer field should be passed on or not (the HTML 5 meta referer tag). By the time of writing, large content promoters rely on the mechanism to pass on some referer information, such as Facebook, Twitter, Bing and Yahoo.

The HTML 5 meta referer tag is however rather recent. Before it, each content promoter used to implement its own solution to pass on the referer to non-HTTPS domains. In Facebook statement, for instance, we see that for browsers not supporting the meta referer tag Facebook first redirects users to a Facebook domain that is still running on HTTP. In this first redirection, the original referer is lost as specified by

---

3.4 Traffic characteristics and impacts on clickstreams detection

![Graph](image-url)

Figure 3.3 Effects of the increasing complexity of web-pages on the percentage of user-actions among HTTP requests.

RFC 7231, but users are still in Facebook systems. Then, users are again redirected to the final destination domain. The final destination receives as referer the HTTP domain under control of Facebook.

As we have seen, the majority of domains was not under HTTPS during the data capture. Popular content promoters were the exception, since they were the early-adopters of full HTTPS support. Since content promoters implement the above techniques to pass on the referer, transitions from the content promoters to HTTP domains are indeed considered. Instead, HTTPS to HTTPS transitions are not visible – e.g., internal navigation on HTTPS domains (e.g., Google→Google) or transition between two HTTPS domains (e.g., Google→Facebook). Clearly, as more and more domains move to HTTPS-only, more transitions become invisible.

3.4.2 Effect of web-page complexity evolution

I investigate how the complexity of web-pages has changed during the data collection. In particular, given my machine learning approach to detect user-actions, I am interested in checking whether key features have varied significantly throughout the years. Major changes in features would affect the performance of the classification
I observe that web-pages have become more complex in recent years. This observation is inline with previous works [52], which report an increasing trend in the number of objects needed to render web-pages. As an example, the median number of children for user-actions is increased by about 40% from 2013 to 2016. As a consequence, the percentage of user-actions among all HTTP requests is decreasing. Figure 3.3 illustrates this effect by depicting the overall percentage of user-actions over time. Observe how similar the trends are across the datasets. The percentage of HTTP requests corresponding to user-actions has decreased from close to 2% in 2013 to less than 1.4% in 2016.

Even if web-pages are becoming more complex, the impact on features relevant to my classifier is limited. As an example, the overall mean number of children per HTTP request is more or less constant between 0.71 and 0.75 in three years, and the vast majority of HTTP requests has no children at all. I remind that the number of children is one key feature used by the decision tree to identify user-actions. Similar observation holds for other features used by the trained decision tree.
3.4.3 Referer artifacts

The analysis of clickstreams depends on the referer field. Previous work [13] has reported artifacts related to lack of referer in HTTP requests. Artifacts related to missing referer are expected to be caused by bugged browser implementations or middle-boxes. The latter is not present in my scenario. I then study the number of HTTP requests that miss referer per browser.

Figure 3.4 illustrates the percentages of missing referer when aggregating all browsers running on PCs (red points). Percentages are between 10% and 15% in the three datasets. This behaviour is consistent if we check different PC browsers in isolation, as well as most browsers running on smartphones and tablets. This small percentage of missing referer is expected. It is caused by normal browsing activity, such as when users request the first web-page after the browser is loaded or when web-pages are loaded starting from bookmarks.

However, a completely different picture emerges for few browsers. Figure 3.4 shows that the percentage of missing referer for a specific Android browser running on Samsung Tablets is much higher than in other cases (green points). This behaviour is restricted to particular versions of the browser. We see that the percentage of missing referer was close to 60% in 2013 and has continuously grown as more users updated to the versions that skip the referer in HTTP requests. I discard such abnormal browsers in the remaining analyses to avoid biases.

3.4.4 Clickstreams on mobile terminals

Some data preparation and filtering is needed to study clickstreams on mobile terminals. Several apps are simply ordinary browsers that behave like PC browsers – e.g., Chrome, Firefox, Safari, Samsung Browser etc. They allow users to move between web-pages and domains and pass on the referer information on each transition. Similarly, many apps include their own browsers and allow users to navigate to other web-pages and domains without leaving the app. This category includes Facebook, Instagram, Flipboard, Messenger, Gmail among others. Here again, the referer information is passed on normally. Each of these apps sends out a customized user agent string and, as such, they are treated as independent browsers given my
definitions (see table 1.1). See section 2.5.3 for the evaluation of the user-action
detection algorithm on mobile applications.

However, several apps constrain users to few operations, and rely on other
browsers (e.g., Chrome) or third-party apps (e.g., Google Maps) to handle external
links. These inter-app transitions are built based on different APIs, and the behaviour
is not standard across apps and mobile operating systems. As an outcome, the referer
information, as observed in the network, when switching between apps is not reliable
– sometimes the referer is an arbitrary string instead of a URL, and often the referer
is simply not present.

I have manually evaluated all popular browsers seen in the traces, and ignored
browsers that do not allow users to navigate through different pages and domains.

3.5 Impact of device on browsing habits

I now focus on the analysis of the clickstream graphs to study the long-term evolution
of browsing habits. The user agent string used (together with anonymized IP address)
for identifying a browser, also identifies a specific application and version and
exposes the operating system and type of device being used. I will coarsely group
browser clickstreams into three classes of browser devices, namely PCs (including
desktops and laptops), tablets, and smartphones.

To illustrate the data used in the remaining sections, I show two examples of
browser clickstream graphs during one day of navigation in figure 3.5. Left graph
refers to a random household in which a PC was used, while the right graph is related
to another random household where a smartphone is seen. Bigger nodes are SEs and
OSNs, and I color them and their neighbors (red for SEs, blue for OSNs). The large
star in the PC graph is due to news from a newspaper web-site. The longer path on the
smartphone is due to content being split into multiple web-pages visited in sequence.
These examples already hint to peculiar topologies and diversity among them. I will
quantify such aspects in the following, taking into account all clickstream graphs in
the dataset.

For simplicity, from now on I report results from PoP 1 only, since the other
probes show analogous results.
3.5 Impact of device on browsing habits

Figure 3.5 Browser clickstream samples for a PC and a smartphone in an arbitrary day. SEs and OSNs (and their neighbors) are marked in red and blue, respectively.

3.5.1 Device usage

I investigate the evolution in popularity of device categories (PCs, tablets and smartphones). I compute, for each device category, its share in terms of number of active browsers, i.e., browsers that generate at least one user-action, and its share in terms of user-actions.

Figure 3.6(a) shows the fraction of active browsers over the last three years. Observe the stunning increase of smartphone browsers, from 26% to 55% of the total active browsers, with PCs that are now less than 40% of the active browsers.

However, contrast this with figure 3.6(b), which shows the fraction of user-actions per device category. Although we see an even larger relative growth in the fraction of user-actions coming from smartphones – e.g., from 7% to 27% – in absolute terms PCs are still creating more user-actions in 2016. For tablets, the increase is more limited, but more visible than in figure 3.6(a) – from 5% to 9%.

Considering the median number of browsers per household, this number has increased from 4 in July 2013 to 7 in June 2016. Smartphones had the largest
increase from 1 to 4 (from 1.5 to 6.4 considering the average). PC category remains constant with a median moving between 3 and 4 throughout the years. Tablets are not widespread and the majority of households does not see any browser of this category (with its mean increasing from 0.4 to 1.2). Remember that when an application is updated, we see it as two distinct browsers.

Concluding, we see an increasing number of custom user agent strings used by different apps in mobile applications, which, recall, we identify as distinct browsers; however, we see a more limited usage of each browser in mobiles as compared to PCs. We next investigate this latter effect in more details.

### 3.5.2 Browsing sessions

**Think-time**

Consider the time between two consecutive user-actions, commonly referred as *think-time*. Figure 3.7 reports Empirical Cumulative Distribution Functions (ECDFs) for smartphones and PCs, comparing July 2013 with June 2016. Tablets are left out to improve visualization.

We observe that in all cases, more than 60% of user-actions are separated by less than 1 minute. The long tail exceeds one day, and peaks are present at typical automatic refreshing time of popular web-pages. Think-time is shorter on PCs than on smartphones, suggesting more interactive browsing sessions on PCs. Think-time slightly increased from 2013 to 2016 (median increased of about 10% for both PCs and smartphones). I have performed a statistical analysis for making claims about differences in distributions. I run a two sample Kolmogorov-Smirnov test with the null hypothesis that the two empirical distributions come from the same distribution. In all cases, the null hypothesis is rejected with a level of significance of 5%. As a counter-proof, sampling from the same empirical distribution, the test does not reject the null hypothesis. I conjecture that this small increase in think-time is related to HTTPS web-pages that we miss more often in 2016, due to the increased use of encryption.
3.5 Impact of device on browsing habits

Figure 3.6 Smartphones present a higher number of browsers, but PCs still dominate the number of user-actions. Browsing from smartphones has increased almost 4 times from 2013.

Session time and activity

I next consider browsing sessions, i.e., the grouped and consecutive generation of user-actions by the same browser. While defining a browsing session is complicated [53–55], I consider a think-time larger than 30 minutes as an indication of the session end. This is a conservative threshold (see figure 3.7), and it is often seen in previous works (e.g., [53]), and in applications like Google Analytics.\(^3\)

Figure 3.7 ECDF of think-time for browsers. Think-time is shorter on PCs than on smartphones.

Figure 3.8(a) shows the distribution of session durations and figure 3.8(b) that of user-actions per session. A session with just 1 action is considered of duration 0 s. Observe that PC sessions last longer and contain more user-actions than smartphone ones. From July 2013 to June 2016 both session duration and number of user-actions per session decreased, with more sessions that include a single user-action – e.g., more than 20% of smartphone sessions have a single user-action in 2016 (in HTTP). The cause for this decrease should be searched in both the increased number of browsers used per household (usage of different apps and also the switching among them) and in the web-pages moved to HTTPS (see next section).

The median number of user-actions per browsing session is small: half of the smartphone (respectively, PC) sessions consist in less than 5 (respectively, 9) web-pages per session, which do not last more than 2 min (respectively, 8 min) in 2016. On the other hand, some few heavy sessions are present: as the tails show, some sessions contain hundreds of web-pages and last many hours.

**Inter-session time**

To complete the analysis, figure 3.9 shows the distribution of idle time between sessions. Results are clearly affected by the periodicity of human life: notice jumps at 24 hours, 48 hours, etc. Also in this case, idle time is shorter on PCs than on smartphones, with median values 3 and 7 hours, respectively. I remind that I am
3.5 Impact of device on browsing habits

Figure 3.8 Browsing session characteristics. Sessions on smartphones last shorter with fewer web-pages than on PCs.

Considering web-page browsing, which is different from other typical usages of mobile terminals, e.g., for instant messaging. Interestingly, smartphone idle time decreased from 2013 to 2016, meaning that the frequency of their usage for browsing the web at home is increasing. The number of times a smartphone browser is used has increased, as well as the number of smartphone browsers.

In a nutshell, people seem to have short sessions on smartphone and tablet browsers, in which they visit a handful of web-pages. Nevertheless, the browsing
intensity is increasing on smartphones (as well as the number of used apps), but PCs are still the preferred means for long, concentrated sessions from home.

### 3.5.3 Content consumption

Next, I quantify the consumption of content per browser and per device category. Figure 3.10 shows the ECDF of the number of web-pages visited per day per browser, considering all days in June 2016. PC, tablet, and smartphone browsers are depicted in separate lines. As expected, the number of web-pages visited on smartphone browsers is significantly lower than the number of web-pages visited on PCs, with tablets in between. This reinforces the observation that smartphone browsers are not used for long web-page browsing at home.

The total daily number of visited web-pages for each browser is in general small: on average, browsers visit 27 distinct web-pages per day on PCs, 15 on tablets, and 10 on smartphones. Only few browsers consume more than 300 web-pages in total. Comparing the number of user-actions (not shown in the figure) to the number of unique visited web-pages, we see that each web-page is visited 1.5 times on average. For this results, figures are overestimated since parameters are removed from URLs. Each browser is seen online only 3.8 days per month (mean values). Recall that households have a median of 7 browsers per month (see previous subsections).
3.5 Impact of device on browsing habits

Figure 3.10 Number of visited web-pages per browser per day. PC browsers visit 3 times more web-pages per day than smartphone ones.

Table 3.2 Details per popular browser application of the average number of visited web-pages per day.

<table>
<thead>
<tr>
<th>Browser type</th>
<th>PC</th>
<th>Smartphone</th>
<th>Tablet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>average</td>
<td>pop.%</td>
<td>average</td>
</tr>
<tr>
<td>Chrome</td>
<td>30.9</td>
<td>46%</td>
<td>9.8</td>
</tr>
<tr>
<td>Safari</td>
<td>23.6</td>
<td>6%</td>
<td>12.4</td>
</tr>
<tr>
<td>Internet Explorer</td>
<td>16.5</td>
<td>20%</td>
<td>-</td>
</tr>
<tr>
<td>Firefox</td>
<td>30.4</td>
<td>20%</td>
<td>-</td>
</tr>
<tr>
<td>Others</td>
<td>22.4</td>
<td>8%</td>
<td>8.4</td>
</tr>
</tbody>
</table>

To give more details, table 3.2 reports the average number of visited web-pages per day, considering popular browser types for each category (PCs, smartphones and tablets). On PCs, the popular browsers account for 92% of the observed browsers, with Chrome being the most popular browser type. On smartphones and tablets, instead, there are many more browser types, so that the ‘others’ class accounts for a significant amount of user-actions. Considering user behaviour on PCs, we observe that Firefox and Chrome users visit more than 30 web-pages per day, while Internet Explorer users are much less active, visiting only 16.5 web-pages. On smartphones, Safari users are more active than Chrome users.

To determine how content consumption is evolving, figure 3.11 shows the evolution of the median number of daily visited web-pages and domains for each browser type.
in the dataset. Results confirm that Internet browsing habits are in general light: the median number of domains visited in each day is pretty low and constant over time.

The median number of web-pages is also limited, never exceeding 27. We observe a decreasing trend for smartphones and, more evident, for PCs. This trend is a consequence of the increasing usage of HTTPS (see figure 3.1). In fact, the figure reports the period in which popular domains started deploying HTTPS (see. figure 3.2). Google and Facebook had already started their migration before July 2013, but there is still a visible reduction in the series for web-pages on PCs, likely connected to the final steps of Google Search migration until September 2013. The migration of some popular domains, such as the Wikipedia, caused a minor, but visible, reduction in numbers for PCs too. The trend is accelerated in the final months of the capture, when other domains starting deploying HTTPS.

In summary, we can conclude that each smartphone browser is used for browsing few web-pages at home, while PC browsers are used for more time and to visit more web-pages. The number of different domains visited over time is typically small and rather constant.
3.6 The clickstream graph

I now characterize properties of the clickstream graphs looking at how people explore the web over time. For each day, I extract and analyze the clickstream graph for each browser (on average, more than 7500 graphs per day), and then compute statistics by aggregating all graphs in each month.

3.6.1 Paths characteristics

First, I gauge how extensive and deep browser explorations are. To reach this goal, I compute how many consecutive and related (by the referer relationship) user-actions, forming a path, browsers visit. More precisely, I extracted the longest among the finite directed shortest paths between all vertices in each graph. Such a path is simply called longest path further in the section. Its length gives a hint on how far the user goes from its navigation starting point over a one-day navigation period.

As seen in the examples of figure 3.5, the clickstream graphs are not random graphs, apparently following a preferential attachment structure, with some hubs with a large number of connected web-pages and some relatively long branches that form the longest paths we are studying.

Figure 3.12 shows the evolution of median and average number of user-actions in the longest paths for PC and smartphones. Paths are quite short, and longer on PCs than on smartphones. Interestingly, path length is stable throughout the years, even if the number of web-pages is reduced (see PCs in figure 3.10). The migration to HTTPS has little impact here: the longest paths do not appear to be exclusively through encrypted domains, even if their popularity (in terms of user-actions) is quite high.

If we count the number of unique domains in each path, we obtain that, on average, only 1.8 domains are present in the longest path. This suggests that people tend to perform deep navigation in web-pages from the same domain, rather than moving among multiple consecutive domains. One could expect this to be a consequence of the fact that most paths are very short. To check whether long paths differ from short paths, I extract path characteristics conditioned to the number of user-actions in the path, i.e., for paths with less than 10 user-actions, from 10 to 100 user-actions, or
Longitudinal characterization of browsing habits

Figure 3.12 Daily number of visited web-pages in the longest path of a clickstream. Depth of the graphs is very limited.

with more than 100 user-actions. Figure 3.13 summarizes with whiskers box-plots the distributions of the number of domains in the paths and it is cut at 6 domains for ease of visualization. Outliers are recognized with the classical Tukey rule on quartile (1.5 IQR method) and shown with diamond markers. In this case, only 3.8% of all the paths are outliers, with the remaining 96.2% of samples that have a number of domains per path that is not greater than 3, with the median number (starred mark) of domains that is always 1 or 2. Therefore the number of domains in paths is always extremely limited, and this number does not increase for longer paths. Unexpectedly, all paths with more than 100 visited web-pages are within 3 domains too. I investigated this behaviour and found that long paths are related to peculiar domains, such as comics or galleries with hundreds of images.

In conclusion, user paths among web-pages are rather short, and rarely users move through many consecutive domains when navigating.

3.6.2 Connected components

I now study the structure of the overall clickstream graph to check whether the several paths forming a clickstream are independent or connected. To this end, I consider the size of the biggest Weakly Connected Components (WCC). A WCC is any maximum sub-graph such that there exists a path between any pair of vertices,
3.6 The clickstream graph

considering undirected edges. In a large WCC, the vertices are connected by edges, i.e., web-pages are visited by following hyperlinks. Recalling the examples of figure 3.5, in those cases we have 4 and 3 WCCs, respectively for the PC and for the smartphone browser, with the biggest WCCs accounting for the majority of the visited web-pages.

As usual, I report the evolution over time of the measurements. Figure 3.14 shows the median of the ratio between the biggest WCC size and the entire graph size, i.e., the relative extension of the WCC. This WCC is a lower bound estimation of the actual WCC: visits to web-pages served over HTTPS are invisible to us; the user-action classifier could miss some user-actions, and some referer fields may be missing in requests. These artifacts correspond to missing edges, thus shrinking the largest WCC. Interestingly, minor changes are observed over the three years, showing minor modification in browsing habits.

Observe that the biggest WCC covers more than 50% of the entire graph, surpassing 70% for smartphones. By manually inspecting the biggest WCCs, I notice that they usually include SEs and/or OSNs, which act as hubs connecting many user-actions. Recalling that paths are usually short this suggests that SEs and OSNs act as starting web-pages used by people to reach other content. I will better investigate this in the next section.

I now focus on those vertices which are not connected to the biggest WCC, but form other (small) WCCs. These typically do not contain SEs or OSNs. A new WCC can indeed be created when there is no referer in a user-action, i.e., when the user...
directly visits a web-page from a bookmark, or by directly entering the address in the browser bar. These would be web-pages users are familiar with, and they reach without following hyperlinks. Notice how the number of vertices that are not part of the biggest WCC is higher for PCs than for smartphones. I conjecture that this can be explained by the different usability of devices: PC browsers facilitate the direct access to content (via both bookmarks, and by auto-completing URLs manually entered). On smartphones, users tend to get support of SEs and hyperlinks to reach the desired content, which results in a more extended WCC.

### 3.7 Content discovery

In this section I investigate in more details how users reach content on the web, and which are the domains that act as content promoters. The out-degree of a domain is the number of hyperlinks leaving web-pages from this domain to web-pages of another one. I use the out-degree of a domain as a metric to pinpoint the promoters: the higher this number is, the more different domains are reached from it. I consider all clickstream graphs in June of each year, and, for each domain, I count the total number of unique domains it appears as a parent at least once.

Table 3.3 shows the evolution of the top promoters in terms of the fraction of different domains they promoted. The great majority are SEs and OSNs. Both
Table 3.3: Ranking (top 8 from 2016) of content promoters popularity with the percentage of domains visited from them.

<table>
<thead>
<tr>
<th>Domain</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
<th>2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>google.it</td>
<td>1 (58.3%)</td>
<td>1 (56.1%)</td>
<td>1 (54.5%)</td>
<td>1 (54.5%)</td>
</tr>
<tr>
<td>facebook.com</td>
<td>3 (6.2%)</td>
<td>3 (6.9%)</td>
<td>2 (8.1%)</td>
<td>2 (9.1%)</td>
</tr>
<tr>
<td>bing.com</td>
<td>5 (2.0%)</td>
<td>5 (2.3%)</td>
<td>4 (2.8%)</td>
<td>3 (4.0%)</td>
</tr>
<tr>
<td>yahoo.com</td>
<td>4 (3.1%)</td>
<td>4 (2.6%)</td>
<td>5 (2.4%)</td>
<td>4 (2.0%)</td>
</tr>
<tr>
<td>google.com</td>
<td>2 (6.5%)</td>
<td>2 (6.9%)</td>
<td>3 (3.0%)</td>
<td>5 (1.7%)</td>
</tr>
<tr>
<td>libero.it</td>
<td>6 (2.0%)</td>
<td>6 (2.0%)</td>
<td>6 (1.4%)</td>
<td>6 (1.2%)</td>
</tr>
<tr>
<td>virgilio.it</td>
<td>10 (0.8%)</td>
<td>7 (1.1%)</td>
<td>7 (0.9%)</td>
<td>7 (1.0%)</td>
</tr>
<tr>
<td>twitter.com</td>
<td>13 (0.5%)</td>
<td>9 (0.5%)</td>
<td>8 (0.6%)</td>
<td>8 (0.6%)</td>
</tr>
</tbody>
</table>

facilitate the user to look for content, and are the typical way to start browsing activities and to discover new domains. Google has a dominant position, with more than 6 times more domains promoted than Facebook which comes second. Its leading position has been constant over years even if in slight decrease, with 55% of domains that have been reached at least one time from google.it domains in June 2016. We observe little changes in the ranking, with Facebook, Bing and Twitter becoming more popular means to reach content.

I now focus on individual clickstream graph to better understand (i) how the usage of SEs and OSNs varies across users and devices; and (ii) if users keep visiting other web-pages after their first visit from a promoter. I manually build a list of the top 50 SEs and OSNs that promote content. To verify to what extent users explore the web after leaving any of these domains, I define the concept of reachability: I say that a destination web-page is reachable if there exists a direct path in the clickstream graph from the promoter web-page to the destination web-page. A web-page is directly connected if it is a child of the promoter. I consider June 2016, and browsers that visited at least 20 web-pages.

Given a browser clickstream graph, consider the fraction of visited web-pages that are directly connected to (or reachable from) an SE or an OSN. Figure 3.15 shows the ECDFs of this fraction across all graphs. For instance, consider the SE curves. 10% of graphs have a fraction of 0%, meaning that 10% of browsers have no
Figure 3.15 Fraction of distinct web-pages per clickstream graph that are directly connected to or reachable from SEs and OSNs. Paths starting from SEs include a sequence of multiple web-page visits, while OSNs typically promote a single web-page.

web-page connected to any SE. The median fraction of web-pages per graph which are directly connected to a SE is approximately 17%, i.e., 17% of web-pages in a browser are found thanks to the direct support of a SE.

Compare now the fraction of web-pages reachable and directly connected to SEs. Figure 3.15 shows that there are many more web-pages that are reachable from search engines (median of 42%) than web-pages directly connected to them. This indicates that users keep browsing the web after leaving a SE, and keep visiting web pages after the initial one. OSNs, on the other hand, exhibit a very different pattern. First, the fraction of web-pages to which they generate visits is much lower, with about 65% of graphs that contains no web-pages visited from any OSNs. Second, the Direct and Reachable curves are very close to each other, suggesting that OSN users visit some external web-page (direct visit), but then do not continue visiting any other web-page (indirect visit). Overall, 61% of the web-pages connected to an OSN have no other child. This shows some “addictiveness” of OSN users: they click on external links, but then get back to the OSN web-page, without continuing exploring the web.

At last, I track the evolution over time of the average direct and reachable fraction of web-pages from SEs and OSNs. Figures 3.16(a) and 3.16(b) report results for PCs and for smartphones, respectively. PC browsers show little changes, with SEs being their preferred means to start their browsing paths and to discover content. OSNs are less often used to discover web-pages, and very rarely to reach web pages not directly
3.7 Content discovery

Figure 3.16 Average number of web-pages directly connected to and reachable from SEs and OSNs. SEs promote more content than OSNs. On smartphones, OSNs have become as important as SEs to directly reach content.
promoted on the OSN itself (cf. figure 3.15). Instead, results for smartphones show a clear evolution over time: OSNs are much more important to discover content than for PC browsers. Even more interestingly, we observe a decreasing usage of SEs to look for content. Notice indeed that the fraction of web pages reached from OSNs is now comparable to the one reached from SEs.

### 3.8 Conclusions

Clickstreams offer invaluable information to understand how people explore the web. This chapter provided a thorough and longitudinal characterization of clickstreams from passive measurements.

My characterization answers two research questions. First, I quantified several aspects of browsing behaviours and how clickstreams have evolved for 3 years. Second, I uncovered properties of clickstream graphs according to the different types of client devices. We observed interesting trends, such as the typical short paths followed by people while navigating the web and the fast increasing trend in browsing from mobile devices. We observed the different roles of search engines and social networks in promoting content in PCs and smartphones. Finally, I also highlighted how the deployment of HTTPS impacts the study of clickstreams from network traffic. My results, while sometimes confirming intuitions, precisely quantify the various aspects of clickstreams, with implications for targeted on-line advertisement, content recommendation, the study of on-line privacy, among others.
Chapter 4

Users’ fingerprinting techniques from TCP traffic

The work I am presenting in this chapter is extracted from my published paper [56].

4.1 Introduction and related literature

Privacy and user tracking are hot topics that impact everyone who uses the web. When online, we offer information about our interests, habits, system configuration, etc., and someone able to eavesdrop the traffic that our devices exchange with the network could invade our privacy. End-to-end encryption – HTTPS – limits access to exchanged information, thus mitigating the problem. Yet, some traffic is transmitted without encryption, such as network and transport layer data.

In this chapter I explore different techniques for profiling and fingerprinting users leveraging the set of domains. I use the term domain informally throughout this chapter, meaning the Fully Qualified Domain Name (FQDN) contacted during web navigation (the definition is slightly changed from the one in the previous chapter, see table 1.1). Indeed, domain names are typically exchanged in clear text, i.e., when resolving a domain via DNS queries because DNS traffic is not encrypted (even DNSSEC does not guarantee confidentiality). Would it be possible to then build a user profile by simply considering the set of domains she visits during her browsing session? And would it be possible to re-identify her in a future time, e.g., when she
is connected in a different network? Real-case scenarios include applications for tracking users in different networks, e.g., tracking users from both mobile and house traffic from a certain area, in which we may want to associate the two datasets. Or, when users change their IPs due to dynamical assignment.

Armed with two large datasets containing traffic summaries of $\approx 7\,500$ anonymized users during 4 weeks in 2017, I answer the previous questions. A big data approach must be considered when creating meaningful fingerprints. I investigate the use of three metrics, considering simple Jaccard index, an information theory Maximum Likelihood approach [57], and a text mining methodology based on TFIDF (Term Frequency - Inverse Document Frequency [58]). I evaluate their performance, highlighting their strengths, weaknesses and trade-offs. Results unveil that TFIDF offers overall the best performance, identifying a given user in different scenarios with up to 94% of accuracy. The rationale of this surprising result is the fact that among the hundreds of domains visited during few days, many are persistent in time and create a peculiar and unique mix of traffic.

The creation of fingerprints to profile users is a problem widely studied in literature under different perspectives. Here I briefly list those works where a limited set of features are considered. Authors in [59] collected volunteers’ web browsing histories (i.e., full URLs), discovering that for 97% of the users, as little as 4 web-pages uniquely distinguish them. Authors in [60] used the DNS traffic to build user fingerprints, reaching accuracy up to 74% in recognition of users in real world traces. Authors in [61] tackle the user tracking and recognition problem exploiting HTTP, HTTPS and SSH protocols, achieving a 50% accuracy. Differently from previous works, I rely on large flow level datasets in which I only consider the name of the contacted server as feature, comparing different methodologies. This work extends the previous work [62], where my colleagues evaluated the usage of the DNS requests to create a fingerprint and track the users over time, discovering that with the simple Jaccard similarity method and one day of traffic only 60% of users can be correctly identified in the future.

To get more insights, I investigate which domains are more useful for such purpose, in particular considering those intentionally visited by the users, that I call Core domains, or those contacted by the browser to fetch objects that compose a web-page or by other background applications, that I call Support domains. To automatically identify them, I propose a methodology based on machine learning,
4.2 Metrics for similarity

and, more specifically, on decision trees, that proved to perform good in the similar scenario of chapter 2. Results show that the intentionally visited web-services prove to better characterize the user than Support domains; however users are better re-identified when all the traffic is taken into account, suggesting that even Support domains help in characterizing users.

My study, although preliminary, shows on the one hand how complicated to protect privacy when online is; on the other hand, the potential of good similarity metrics and machine learning applications linked to, e.g., forensic. To foster new studies and permit results reproducibility, I contribute also this dataset to the community. Anonymized sets of domains are available to the public at http://bigdata.polito.it/content/domains-web-users.

4.2 Metrics for similarity

In this section I provide a description of the three different methodologies upon which I base my profiling and identification techniques. Consider a user $u \in U$ in the collection of users $U$. In a time period $\Delta T_1$ she visits a set of domains $D_{u,\Delta T_1}$. I will call $\hat{D}$ the set of all the domains seen by the population at time $\Delta T_1$, i.e.,

$$\hat{D} := \cup_{u \in U} D_{u,\Delta T_1}.$$ Given the profiles $D_{u,\Delta T_1}$ for all $u \in U$ in a certain $\Delta T_1$, I suppose to have the set $D_{v,\Delta T_2}$ in a future $\Delta T_2$ only for user $v$, but without knowing the identity of user $v$ in $\Delta T_2$. My goal is to correctly identify the user $v$ among the profiles of users in $U$ built in the past. The underlying hypothesis is that there is a positive-correlation among the domains retrieved by a user in different time windows.

4.2.1 Jaccard Index

The Jaccard index measures similarity between two finite sets. It is defined as the size of the intersection divided by the size of the union of the two sample sets:

$$Jac(D_{u,\Delta T_1}, D_{v,\Delta T_2}) := \frac{|D_{u,\Delta T_1} \cap D_{v,\Delta T_2}|}{|D_{u,\Delta T_1} \cup D_{v,\Delta T_2}|} \quad (4.1)$$

Jaccard index is in $[0,1]$, and it is equal to 0 when there is no common element between the two sets, while if the two sets contain same elements, the index equals 1.
4.2.2 Maximum Likelihood Estimation

For this method, I assume a simple behavioural model for the visited domains in which a user’s likelihood of visiting a certain domain is governed by the domain overall popularity and whether this domain already appeared in her previous domains set. Then, for each user \( u \in U \), we compute her likelihood of generating the domain set \( D_{v,\Delta T_2} \) under the imposed model. This method, with few modifications, has been already proposed in [57], where the proof can be found.

I suppose that each domain \( d \in \hat{D} \) has a certain likelihood \( p(d) \) of being picked in \( \Delta T_2 \), independently from the user, given its popularity in \( \Delta T_1 \):

\[
p(d) := \frac{|\{ u : d \in D_{u,\Delta T_1} \}|}{|U| \cdot z_1}
\]

where \( z_1 \) is a normalization factor.

We can suppose a user is more likely to visit a domain she already visited in the past. For any \( u \in U \) and parameter \( r \geq 0 \) we define a random variable \( H(u,r) \in \hat{D} \) s.t.:

\[
Pr(H(u,r) = d) := \begin{cases} 
    r \cdot p(d) / z_2 & \text{if } d \in D_{u,\Delta T_1} \\
    p(d) / z_2 & \text{otherwise}
\end{cases}
\]

(4.2)

where \( z_2 \) is a normalizing factor. Then, given a number equal to \( |D_{v,\Delta T_2}| \) of i.i.d. draws of \( H(u,r) \), the Maximum Likelihood Estimation \((\hat{u}, \hat{r})\) of the underlying parameters for producing the set \( D_{v,\Delta T_2} \) are:

\[
\hat{u} = \arg \max_{u \in U} \left\{ q_u \log \frac{q_u}{s_u} + (1 - q_u) \log \frac{1 - q_u}{1 - s_u} \right\}
\]

\[
\hat{r} = \left( \frac{q_{\hat{u}}}{1 - q_{\hat{u}}} \right) / \left( \frac{s_{\hat{u}}}{1 - s_{\hat{u}}} \right)
\]

where \( q_u = |D_{u,\Delta T_1} \cap D_{v,\Delta T_2}| / |D_{v,\Delta T_2} \cap \hat{D}| \) and \( s_u = \sum_{d \in D_{u,\Delta T_1}} p(d) \). \( q_u \) is the fraction of domains of \( D_{v,\Delta T_2} \) that are in a previous set \( D_{u,\Delta T_1} \); \( s_u \) is the generalized size of \( D_{u,\Delta T_1} \), where it accounts both for the total number of domains in \( D_{u,\Delta T_1} \) and the popularity of those items. Intuitively, \( \hat{u} \) is a user for which \( q_u \) is large and
s_u is small; that is, D_u, \Delta T_1 is not too big, but contains many of the domains in the observed history D_v, \Delta T_2. The model allows for r < 1, in which case D_u, \Delta T_1 is an anti-recommendation set. However, I consider here only the cases where r > 1. If such r does not exists, we consider \hat{u} for which r is bigger.

### 4.2.3 Cosine similarity based on TFIDF

TFIDF is the product of two statistics, Term Frequency and Inverse Document Frequency, and it is widely used in information retrieval. TFIDF reflects how important a domain d is for a user u \in U, with respect to the set of all users U.

Term Frequency (TF) measures the importance of domains for user u. Differently from the classic version of TF, since we are using just sets of domains it will be independent of a particular domain:

\[
TF(u) := \frac{1}{|D_{u, \Delta T_1} \cap \hat{D}|}
\]

IDF measures how important a domain is in the whole collection of users, in \Delta T_1. While computing TF, all domains are considered equally important. However certain domains may be very popular and therefore have little importance. Thus, we weight less the frequent domains while scale up the rare ones.

Notice that for a user in \Delta T_2, we are removing the domains never seen by the population at time \Delta T_1, because such new domains will not have an Inverse Document Frequency (IDF) – remind that \hat{D} is built at \Delta T_1.

IDF is defined as:

\[
IDF(d) := \log \frac{|U|}{|\{u : d \in D_{u, \Delta T_1}\}|}
\]

Hence \(IDF(d)\) is the logarithm of the total number of users divided by the number of users having seen domain d in \Delta T_1. Finally, TFIDF is the product of the two, i.e., \(TFIDF(d, u) := TF(u) \cdot IDF(d)\). Then we can compare how similar two domain sets D_u, \Delta T_1 and D_v, \Delta T_2 are, computing the cosine distance of the two multi-dimensional arrays:
$\text{Cos}(D_{u,\Delta T_1}, D_{v,\Delta T_2}) := \sum_{d \in D} \text{TFIDF}(d,u) \cdot \text{TFIDF}(d,v) / ||\text{TFIDF}(d,u)|| \cdot ||\text{TFIDF}(d,v)||$

with $||\cdot||$ indicating the usual Euclidean norm. The resulting similarity ranges from 0, meaning no elements in common, to 1, meaning all the entries being exactly the same.

### 4.2.4 Complexity

Assuming to compute a similarity between a single user and other $M$, each with $N$ domains, Jaccard index computation costs at most $O(M \cdot N^2)$ time. If $P$ is the total number of domains seen by all the $M$ users, with $N \leq P \leq M \cdot N$, TFIDF and MLE methodologies cost at most $O(M \cdot N \cdot P)$ time. This is due to the fact that TFIDF and MLE make comparisons in the larger set of all domains.

Hashing such as MinHash could be used on top of my computation to speed-up the process. With a smart implementation using hash functions, the computation cost could decrease to $O(M \cdot N)$ for Jaccard and $O(M \cdot P)$ for MLE and TFIDF.

In the user tracking problem, $N$ should be relatively small (up to few thousands domains), while $M$ and, by consequence, $P$, could range from a few users to millions of them, depending on the application. In case of very large $M$, it is therefore much faster to use the simpler Jaccard similarity.

### 4.3 Identification of Core domains

When visiting a web-page, the browser application first downloads the main HTML document and then fetches all the objects of the page (images, scripts, advertisements, etc.). These are often hosted on external servers (e.g., CDNs) having different domains. We call Core domain a domain originally contacted to download the main HTML document of a page. Core domains are important since they are intentionally visited by users, like www.facebook.com and en.wikipedia.org. We call Support domains those domains automatically contacted by visiting a Core domain, or by background applications, like static.10.fbcdn.net and dl-client.dropbox.com. Support domains do not contain useful information
4.3 Identification of Core domains

about user intention. When analyzing network traffic, having a list of all possible Core domains is important to make user behaviour emerge. Are Core domains important also for user tracking and identification?

In the literature, previous works tackled the problem of identifying intentionally visited domains, also called user-actions. See chapter 2 for more details. However, an important fraction of traffic is becoming encrypted, making the approaches of chapter 2 less effective and leaving passive probes with no visibility on HTTP fields. Few efforts have been put in identifying user actions from encrypted traffic [63, 64]. These works aim at identifying user-actions from flow level measurements examining traffic at runtime. Here, on the contrary, we aim at building a list of domains that typically contain user-actions since they host actual web services.

Given a domain, we visit the home-page it hosts. Based on the response, we classify it as a Core or Support domain. This is a classification problem, that we solve with a machine learning approach using a decision tree classifier (instead of building a manual custom heuristic to solve the problem).

First, we need to define the set of features to use: we consider an extensive list guided by domain knowledge, and let the classifier choose the ones that better allow us to separate Core and Support domains. Features include the length and the content type of the main HTML document (if present); the number of objects of the page and domains contacted by the browser to fetch all objects; HTTP response code (e.g., 2xx, 3xx and 4xx); and whether the browser has been redirected to an external domain. To get the set of features, we use active crawling, and visit the home page of each domain by means of Selenium automatic browser to extract page features.\(^1\)

To train the classifier, we build a labeled dataset that we use for training and testing, considering a list of 500 Core and 500 Support domains. More in detail, we picked the list of domains found in the Campus trace (see next section for details), sorted by number of visits. Then, we manually visited each home page corresponding to the domain name. We manually label such domain as a Core or a Support domain by looking at the rendered web-page. We stop the labeling process when we reach 500 items for each class. We obtain a balanced labeled dataset that we make publicly available at http://bigdata.polito.it/content/domains-web-users. For the decision tree, we opt for the J48 implementation of the C4.5 algorithm offered by Weka.\(^2\)

---


Interestingly, the final decision tree, which I report in figure 4.1, results in a very simple, efficient, and descriptive model which reads as: a) the main HTML document size must be bigger than 3357B and b) the browser must not be redirected to an external domain.

Intuitively, Support domains typically lack of real home page. When directly contacted, the server reply with short error messages. In some cases, Support domains redirect visitors to the service home page (e.g., fbcdn.net redirects on www.facebook.com).

Despite its simplicity, overall accuracy is higher than 96% when tested against 1000 labeled domains, using 10–fold cross validations.

The methodology introduced in this section was primarily designed by my colleague Ing. Martino Trevisan (more information at https://www.tlc-networks.polito.it/public/phd-and-post-docs/martino-trevisan). The methodology will be also used in chapter 5.

### 4.4 Datasets for experiments

For my analysis I mainly rely on a dataset collected in my university campus in Torino, where I consider the traffic of approximately 2500 users during 4 weeks between January and February 2017. Users are faculty members whose terminals are directly connected to the Internet via wired Ethernet, using fixed IP addresses that I
Table 4.1 Overview of the considered datasets: traces and characteristics.

<table>
<thead>
<tr>
<th>Trace</th>
<th>Log Size</th>
<th>Volume</th>
<th>Client IPs</th>
<th>Domains (FQDN)</th>
<th>SLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Campus</td>
<td>229 GB</td>
<td>113 TB</td>
<td>≈ 2 500</td>
<td>404 k</td>
<td>136 k</td>
</tr>
<tr>
<td>ISP</td>
<td>440 GB</td>
<td>232 TB</td>
<td>≈ 5 000</td>
<td>611 k</td>
<td>204 k</td>
</tr>
</tbody>
</table>

use as identifier of the terminal itself. Hence I assume each IP address is associated with one and only one user.

I rely on Tstat to perform passive measurements (see section 1.2). Here, we are interested in retrieving the domain of the server being contacted. For plain HTTP flows, the Host header is extracted from HTTP request. In case of TLS, the DPI module provides the Server Name Indication (SNI) field in Client Hello message. SNI is a TLS extension by which the client indicates the domain of the server that is trying to contact. At last, it reports the domain name clients resolved via DNS queries prior to flows [7]. I combine these three sources to label each flow with the server name, giving higher priority to Host and SNI fields where more than one is present.

First row of table 4.1 summarizes the characteristics of Campus dataset. In total, I observed about 404 k unique domains, corresponding to more than 136 k unique second-level domains. In total, 691 millions flows have been observed during 4 weeks between January and February 2017; no holidays or festivities occurred during this period. I load and process the logs using Apache Spark in a 20-machine Hadoop cluster, capable of reading from disk and processing the Campus dataset in about 20 minutes. With the same machine, it took about 1 hour for classifying 404 k domains as Core or Support domains. The Internet access speed (1Gbps in this case) was the main bottleneck.

Figure 4.2 shows the number of unique domains (Core and Support domains) seen in the whole dataset. New domains are continuously discovered, reaching ≈ 404 k after 4 weeks, of which Core domains are ≈ 115 k (left y-axis). Next, we observe the number of active users creating some traffic for each hour; they vary from 400 at night to almost 1,400 at midday. The median over the users of unique domains discovered over time (right y-axis) is shown. Ratio between Support and Core domains increases, with a median of 662 Support and 70 Core domains after 4 weeks. Domain popularity is unbalanced: a very small portion of domains are
popular and just 634 domains out of the 400000 are contacted by 10% or more of the users. Core domains are about 32% of the total; however, only 4% of the top 1000 most popular domains are Core domains.

The dataset here discussed will be also used in chapter 5.

Finally, in the experiment in section 4.5.5 I use a second trace. I deployed Tstat during the same period in a PoP of an European ISP where the traffic of ≈5000 ADSL and Fiber households is aggregated. All households have a fixed IP address which identifies a home gateway acting as NAT router, behind which multiple user devices may be connected to the Internet. Therefore a single identifier (i.e., the client IP address) can actually hold more than one user. Second row of table 4.1 reports the dataset statistics.

### 4.4.1 Ethical and privacy implications

Both the data collection process and the collected data have been discussed, reviewed and approved by the ethical board of my university. I took all possible actions to protect leakages of private information from users. In particular, I anonymized the IP addresses of clients using a technique based on irreversible hash functions, only retaining the data that is strictly needed for my study. More in detail, I limit the data to i) the anonymized client IP address, ii) the name of the contacted server, iii) the timestamp of the TCP connection. I further anonymized the server name in the
4.5 User fingerprint and identification

datasets that I contribute to the community which contains data collected from my campus.

Considering the data collection process in the ISP, the same precautions have been implemented, and the data collection process has been reviewed and approved by the ISP security board. In this second scenario, I have no information at all about the ISP customers.

For more details on data collection and ethics see also appendix A.

4.5 User fingerprint and identification

In this section, I show different case studies for defining a user fingerprint. Our main task is to identify a user, by tracking her visited domains. In the first phase, I profile the users creating fingerprints, while in a second phase, I try to identify a given user in a later trace. All clients are provided with a public and fixed IP address that I use as identifier to build the ground truth. I consider 1,205 users that were active during the period of time of data collection. The constraints used for choosing the 1,205 users are written in each experiment. When otherwise stated, the 1,205 users are chosen randomly. Performance of each similarity metric is measured by the percentage of users correctly identified. The results can be meaningfully compared since they are always related to 1,205 possible cases.

4.5.1 Role of number of domains

The first set of results aims at assessing the importance of the number of Core and Support domains for the profiling and identification tasks. I collect the domains visited by users and generate a profile $D_{u,\Delta T_1}$ for each $u \in U$. In the next step, I associate user $v$ with profile $D_{v,\Delta T_2}$ to the most likely user $u \in U$. If $u = v$ we have a positive match. Figure 4.3 shows the percentage of users that were correctly identified, versus the number of Core and Support domains used in the profile generation phase. For equal user number, the larger is the data, the better is the identification. For low number of both Core and Support domains, it is hard to identify users; in fact, just up to 23.8% were correctly identified with 10 domains.
Core domains are clearly better characterizing and more important for user identification than Support domains, for all the three metrics. With only 70 Core domains, the range of correctly identified users is equal to 72-79%. This percentage is higher than the range of correctly identified users when 500 Support domains are used (68-73%). Regarding to the performance of the three similarity metrics, Jaccard performs worst in all the cases; TFIDF has the best results in most of the experiments with Support domains, while MLE performs a slightly better with Core domains.

### 4.5.2 Role of observation time

I now focus my attention to study how observation time impacts on profiling and identifying users. The rate for discovering new domains differs from user to user, and I expect that the same number of domains is collected after a different amount of time. For instance, for discovering 50 Support domains it takes, in median, only 11 hours. For Core domains, instead, it takes about 3.5 days. 5 days are needed to obtain 70 Core domains, but only 3 days for 500 Support domains. This is mainly due to the large number of Support domains (see section 4.4).

Here, I maintain constant the time of observations $\Delta T_1$ and $\Delta T_2$. I consider two consecutive weeks of traffic, taking into account only users that visited at least 20 Core domains per week, avoiding those that either disappeared or generated little traffic. I use the first week for profiling, and the second one for identification. Table 4.2 details results for the three similarity metrics considering Core, Support
4.5 User fingerprint and identification

and All domains. The median number of domains per week per user is reported for completeness. Best results are in bold. I repeat the same experiment considering 2 consecutive days, with results in table 4.3, where the median number of discovered domains decreases to about the half, and performance also drops. In both cases the large quantity of Support domains, about ten times the number of Core domains, helps in user identification.

As expected, performance decreases when scarce information is available, in particular moving from the two weeks experiment to the two days one. However, still 70% of users can be identified when observing them during one day only. Results of the three metrics are consistent to the ones showed in figure 4.3: Jaccard has always the worst performance, MLE seems less affected by the limited size of Core domains and TFIDF offers the overall best performance except when few Core domains are considered.

To give an intuition about the discriminative power of the profiles, I report in figure 4.4 the Cumulative Distribution Function (CDF) of TFIDF metric between the same user (called self-similarity) and between two different users, considering all the domains from the same two consecutive weeks of table 4.2. Self-similarity is almost always higher than the similarity with a different user, thus allowing us to correctly identify the target user.
Table 4.2 Identified users in two consecutive weeks.

|          | Median $|D_{u,\Delta T}|$ | Jac  | MLE  | TFIDF |
|----------|----------|------|------|-------|
| All      | 710      | 76.9%| 79.6%| 82.3% |
| Core only| 69       | 67.2%| 70.9%| 70.8% |
| Support only | 641 | 75.4%| 78.3%| 80.7% |

Table 4.3 Identified users in two consecutive days.

|          | Median $|D_{u,\Delta T}|$ | Jac  | MLE  | TFIDF |
|----------|----------|------|------|-------|
| All      | 325      | 63.2%| 67.6%| 71.2% |
| Core only| 26       | 45.8%| 55.1%| 50.1% |
| Support only | 294 | 61.2%| 65.6%| 70.3% |

4.5.3 Longer profiling or identification?

Differently from the previous results, here I consider all users in the dataset, allowing real cases in which users may disappear and generate little traffic. I consider two cases. First, I assume to have i-a) fixed and long profiling time, i.e., 2 weeks for each user, and ii-a) variable observation windows for identify them. Second, I assume to have i-b) a fixed amount of 2 weeks for identification, but ii-b) variable time profiling all the users. Figures 4.5(a) and 4.5(b) depict the results for both aforementioned cases. Jaccard considers the two sets independently from the rest of the population; therefore such metric is symmetric with respect to the two sets and depends just on their sizes. This implies that Jaccard performs equally in a) and b) cases, reaching good performance only when the training and testing sets of domains are both large. TFIDF and MLE metrics account for the whole population when profiling, hence having large profiling sets for the population is much more important than having a single large identification set. This is why with a 2 weeks profiling time TFIDF and MLE show good performance even with just few hours of identification time. On the contrary, building profiles with few hours of traffic makes the large amount of data for identification much less useful.

In a nutshell, building better profiles for the whole population is much more important that having a lot of data for identification.
4.5 User fingerprint and identification

4.5.4 Profiling aggregate traffic

Here I consider to build a profile from different groups of users, and then see if it would be possible to associate a user to her group. For this analysis, I consider the aggregate traffic produced by all people in the same University Department during one week, and build a profile for the active department. Medianly, each department profile contains \( \approx 24 \) k domains, two orders of magnitude more than single user case. Next, I consider each user (from the following week of traffic) with the goal of assigning her to the correct department. Having my Campus 10 departments, a random unbiased method assigns correctly 10% of users.

Results show that Jaccard performs poorly, correctly associating only 8.7% of users: without weighting the popularity of domains, the traffic of the single user it is almost indistinguishable within the departments. TFIDF instead reaches a surprising 73.9% accuracy. Intuitively, TFIDF is able to identify the peculiar, per-department, domains, i.e., those with high IDF among the huge quantity of domains. MLE reaches 23.4%; its performance are biased from the fact that we assume a user will contact web-sites according to the model of Eq. 4.2, based more on the persistence of the traffic than on the peculiarity of domains for each user.

This experiment suggests also that users in the pool are quite similar, and thus harder to identify, e.g., it is easy to distinguish a electrical engineering department user from an architecture department professor, but harder to distinguish two electrical engineering department users.
Table 4.4 Users correctly identified in residential dataset.

|        | Median $|D_{u,\Delta t}|$ | Jac  | MLE  | TFIDF |
|--------|----------|------|------|-------|
| 1 day  | 556      | 80.4%| 84.1%| 86.6% |
| 1 week | 1785     | 93.6%| 93.7%| 94.9% |

4.5.5 ISP Case

I now repeat the experiments of tables 4.2 and 4.3 using the ISP trace, where users are possibly more heterogeneous. To compute results against the Campus Dataset, I randomly select 1,205 users, among the active ones. I profile them for one day (respectively, week), and identify using data from the second day (respectively, week). Results are in table 4.4. Recall that here we have residential access, with possibly multiple devices multiplexed on the same public IP address that I use as identifier. This factor contributes to explain the high median number of domains. Identification accuracy tops to more than 86% (respectively, 94%) with TFIDF, quite a surprising result. This, other than the bigger amount of data, is also due to the more heterogeneous navigation habits of residential users with respect to campus ones. In fact, 1 day of profiles in the ISP dataset has less median number of domains than 1 week of profiles in the Campus dataset (see table 4.2), but reaches better performance.

4.6 Conclusions and discussion

I explored techniques for users’ fingerprinting and identification using only the domains of visited web-services. Results show that a simple approach like the TFIDF can be used to solve the identification problem in different scenarios, provided users are profiled for enough time. Web-services intentionally requested have proved to better characterize users.

I expect the probability of identification to decrease with respect to the user population size. My conclusions could be over-optimistic with larger user-base, and I am planning to repeat the same experiments with larger datasets. I am also focusing on extending this approach to include more features e.g., the information coming from timing, or volume of data.
Chapter 5

Mining and modeling web trajectories from TCP traffic

The work I am presenting in this chapter is extracted from my published paper [65].

5.1 Introduction

Understanding how people move within web-sites has been always a important problem [26] for a variety of purposes like recommending content [4], comparing rankings in the web [30], or increasing privacy and security [5]. Users’ browsing activities can be described by means of the paths that they follow when navigating through web-sites. The evolution of the web, obviously, changes how users interact with it.

As shown in chapter 4, even with nowadays widespread encryption at the application layer, a passive observer of the network can still obtain valuable information about the trajectory a user follows in the web. For instance, the domains of the web-sites contacted during browsing are still not encrypted and easily accessible from passive probes. Indeed, domains are exchanged in clear text, i.e., when resolving a domain via DNS queries [7]. In this chapter I will use the term domain to refer to the combination of second level domain and top level domain, as done in the previous chapter.
In this work I refer to the sequence of domains visited by a user as the user trajectory. Both user’s circumstances and preferences affect such trajectories. Here, I consider only the domains intentionally visited, i.e., Core domains. Finding those in the stream of all contacted domains requires ingenuity, as I showed with the methodology introduced in section 4.3.

In figure 5.1 there is an example of a trajectory on the web. Notice how the same domain can appear multiple times.

Armed with these sequences of visited domains, i.e., user’s trajectories, I analyze them using TribeFlow, a methodology that the professors I was working with (Prof. Flavio Figuireido et al.) originally proposed in 2016 [8, 66] to model each user as a random surfer over latent environments. User trajectories are the outcome of a combination of latent user preferences and the latent environment that users are exposed to in their browsing.

I build this model and analyze its results using the large dataset presented in section 4.4, containing traffic summaries of ≈ 2 500 anonymized users in my university campus in Torino during 4 weeks in 2017. A big data approach must be considered for retrieving, processing and managing such amount of data.

Thanks to the model, I show how to automatically group together the interests and browsing patterns of single users and/or communities into so-called environments. I propose an automatic way to highlight differences in terms of popularity and content. Lastly, I analyze the transition among environments, showing how the single department has a much more homogeneous behaviour than the whole university.

My analysis shows that it is possible to:
model accurately the users trajectories, by simply considering domains names, without the need of the whole packet traces;
• extract environments with similar or likely connected web-sites;
• highlight differences in terms of popularity and content of environments;
• extract the interests of communities of people.

The workflow of the system is the one sketched in figure 5.2. Users are connected to the Internet and I monitor the network at a point of presence where I collect and log information about each TCP connection. From this log, I extract all the domains, for each user. I use the methodology explained in section 4.3 that, thanks to some active measurements, get the subset of domains related to services intentionally visited by users. I (i) focus on those domains, and (ii) reconstruct meaningful trajectories over time (section 5.2). At this point, I learn the possibly best fitting TribeFlow model on such data (section 5.3) and I analyze the results (section 5.4).

To foster new studies and permit results reproducibility, I contribute the dataset and model to the community. Anonymized trajectories of domains and their models
are available to the public at http://bigdata.polito.it/content/domains-web-users, while TribeFlow code is available at https://github.com/flaviovdf/tribeflow.

5.2 Trajectories reconstruction and characterization

First, I characterize the trajectories in the dataset. These results will support the credibility of trajectories and therefore the model assumptions, as I will describe in section 5.3.

I recall that the dataset (section 4.4) contains traffic of approximately 2,500 terminals. Here, for each TCP connection I just consider: (i) the anonymized client IP address as terminal identifier, (ii) its department inside the Campus, (iii) the starting time and (iv) the end time of the connection, and (v) the server domain name.

Again, I apply the classifier of section 4.3 to the dataset, extracting the Core domains. Figure 5.3 depicts two curves of domain popularity, where I rank each domain according to the fraction of visits directed towards it, for both all and Core domains. Notice the loglog scale. The pattern is similar for both the curves, even if the number of domains is quite different: a small portion of domains are popular and get the great majority of visits, with a long queue of domains getting few visits. Among the top 1,000 most popular domains (all), only 61 are Core domains. For Core domains, the popular domains are basically the ones with most page-views, according to Alexa, i.e., Google, Facebook, Youtube, etc. These bring little information on the specific interests of a user. On the contrary, I expect the less popular domains to give specific information about user interests.

It is also interesting to analyse the connection length and inter-arrival time distributions. Both measures are important given that I want to reconstruct meaningful trajectories based on relative arrival time of the visit. Figure 5.4(a) reports the empirical Cumulative Distribution Function (CDF) of the connection duration. Core connections are in general longer. 80% of the connections last less than 3 minutes. The inter-arrival between two consecutive connections is shown in figure 5.4(a). It is in general shorter than the duration of a connection. This means that is not uncommon to have multiple contemporary connections. This could imply that there

are contemporary connections towards multiple sites and therefore could be complex to estimate the sequence of visits of the user. Moreover we see that many connections have a very small inter-arrival time ($\leq 0.01$ s). This is due to the fact that browsers open multiple connections when visiting a web-page. These connections could be towards many servers (Support domains), or even multiple connections towards the same domain.\footnote{\url{http://www.browserscope.org/?category=network&v=top}, accessed January 2018.} I analyze this aspect by showing the number of overlapping connections. Figure 5.4(b) shows the probability of number of overlapping connections. The x axis is limited to 6 for ease of visualization. Analyzing all the domains, multiple contemporary connections are usual, and even tens of them are not rare. Even when limiting to Core domains, the probability of having no contemporary connections is only 26%.

When building the user trajectory, multiple contemporary connections towards the same domain would artificially create sequences of visit to the same domain. To filter this artifact, I group together such overlapping trajectories towards the same Core domain. Specifically, I group visits to the same domain happening in the same time window of duration equal to the duration of the first connection. I then recompute the number of concurrent connections, showed by the curve labeled \textit{Core - filtered} in figure 5.4(b). Almost no connection is now contemporary to others (less than 9%).
I want to model user habits focusing on how they move in the web from domain to domain. To achieve this goal, I rely on the recently proposed model called TribeFlow [8, 66]. The methodology summarized in this section was primarily designed by Prof. Flavio Figueiredo (more information at http://www.dcc.ufmg.br/dcc/?q=pt-br/node/2477), with whom I am collaborating. Here I provide a brief and simplified description of the methodology and invite to refer to [8] for all details.

I represent the trajectory of each user \( u \in U \) as a sequence of requests to domains. \( U \) is the set of users in the whole trace \( V \). Similarly, \( d \in D \) represents a domain. We generalize the dataset \( V \) from a single edge network as a set of visits \( v \in V \), with each visit being represented as a triple: \( v = (u, t, d) \). \( t \) represents a timestamp of the visit.

5.3 Modeling user trajectories with TribeFlow

(a) Duration and inter-arrival of connections

(b) Number of contemporary visits

Figure 5.4 Characterization of connections.
Figure 5.5 Each user performs random walks over latent environments, with different probability of interest (preference strength) towards environment.

It is expected that real user behaviour will be (i) non-stationary, and (ii) time heterogeneous [67]. In other words, user behaviour change and evolve over time, and is different for each user. TribeFlow was designed to cope with the complex challenges of learning personalized predictive models of non-stationary, time heterogeneous, and transient (Markovian) user trajectories.

Let a trajectory from a users be comprised of the visits ordered by time. That is, the trajectory for user $u$ is: $\mathcal{V}_u = (u, t_0, d_0), (u, t_1, d_1), \cdots, (u, t_n, d_n)$. TribeFlow objective is to capture the overall set of trajectories as a set of $k$ transitions matrices of size $|\mathcal{D}|$ by $|\mathcal{D}|$. Each matrix represents a stationary and time homogeneous first order Markov chain and characterize a latent environment $z$. Therefore, each environment is associated with the first order Markov chain $P(d_{i+1} | d_i, z)$. The domain $d_i$ is associated with time $t_i$, whereas $d_{i+1}$ is the next immediate visit at time $t_{i+1}$. This chain captures the probability of going to $d_{i+1}$ given that the last visit was to domain $d_i$, within environment $z$. The number of environments $k$ can be automatically inferred from the data with different methods [66].

For each user $u$, I define a probability of interest, or strength, that this user has in environment $z$, $P(z | u)$. That is, when a user is browsing the web, she selects a environment associated to her interests or needs following a distribution (e.g., search engines, social networks, or e-commerce web-sites). This choice is done according to $P(z | u)$ at every step. After selecting an environment, the user selects a domain based on $P(d_{i+1} | d_i, z)$. 
Based on the definitions above, the whole dataset of trajectories are captured as random walks over random environments. Each environment captures a latent factor that leads to a user visiting a domain. See the representation in figure 5.5.

Given that for each user the overall behaviour is captured as a mixture (based on $P(z | u)$) of stationary and time homogeneous matrices ($P(d_{i+1} | d_i, z)$), the overall behaviour captures a non-stationary, time heterogeneous, and transient system. Finally, TribeFlow can non-parametrically incorporate the inter-arrival time to model $P(\tau | z)$, with $\tau = t_{i+1} - t_i$. The overall probability of a user transitioning between two domains within an environment is defined as:

$$P(d_{i+1} | d_i, u, z, \tau, \alpha, \beta, \lambda) = \frac{P(d_{i+1} | z, \beta)P(d_i | z, \beta)P(\tau | z, \lambda)P(z | u)}{1 - P(d_i | d_i, z, \beta)}$$

$\alpha, \beta$, and $\lambda$ are hyper parameters which are fixed to sensible defaults [8]. The goal of the model is to learn the matrices $\Theta$ (with $k$ rows and $|U|$ columns) and $\Phi$ (with $|D|$ rows and $k$ columns) that correspond to probabilities $P(z | u)$ and $P(d | z)$, respectively. In $\Theta$ each cell captures $P(z | u)$ for a given environment and user. After the model is trained, the probability $P(d_{i+1} | d_i, z)$ is simply defined as: $P(d_{i+1} | d_i, z) = P(d_{i+1} | z)P(d_i | z)$ The probability matrix for each environment, as well as the strength of each user in an environment is learned through Gibbs Sampling (see [8] for details).

**Learning the model on the dataset:** using TribeFlow, I am able to summarize the large dataset of trajectories in a succinct and interpretable manner. For training the model, the only parameter I have to tune is the number of environments $k$. More environments allows better approximating the trace, but increase the complexity of the models. I therefore use a weighted version of the error, obtaining a best number of environment $k = 30$.

I train different models, each with 30 environments, by using: 1) the whole Campus trace, 2) the single Architecture department trace and 3) the Electronic department trace.
5.4 Results

In this section, I analyze the model I created in section 5.3 and I show how it can be useful for studying the behaviour of the users.
5.4.1 Environments as clusters of common domains

Once I build the 30 environments, I manually inspect and analyse the most relevant domains inside them. We can expect each domain $d$ to be present in all the environments $z$, with a probability $P(d|z)$. We can therefore sort the domains $d$ accordingly to the probability distribution $\Phi_z(d) = P(d|z)$ inside each environment, and then visually inspect the most probable domains. By construction, I expect to have domains that are usually sequential in time within the same environment. Results show that most of the environments contain the most popular and generic domains, e.g., search engines and social networks. However, we clearly see some specific domains to dominate in some environments, i.e., the model let domains with the same topic to emerge in some specific environment only. For instance, we observe news, weather forecast, engineering forums and science journals related
5.4 Results

Figure 5.8 Transition probability heat-maps among the environments.

web-site to dominate different environments. This reflects the user habits of browsing a sequence of domains with the same topic. To illustrate this, consider the models extracted from users within the Electronic and Architecture departments. Figure 5.6 illustrates the word-clouds of the top 10 domains (just second level domains depicted) of specific environments. I removed common domains, i.e., Google, Facebook, and Youtube because they are among the most popular in all the environments. Observe how expressive are the word-cloud in describing the topic of each environment.

5.4.2 Analysis of diversity and popularity of environments

To quantify how the 30 environments are different among each other, I use as metric the Kullback-Leibler (KL) divergence [68]. KL divergence \( D_{\text{KL}}(\Phi_{z_1} \parallel \Phi_{z_2}) \) measures how one probability distribution \( \Phi_{z_1} \) diverges from a second, expected probability distribution \( \Phi_{z_2} \). It is defined as:
\[
D_{KL}(\Phi_{z_1} \parallel \Phi_{z_2}) = \sum_{d \in D} \Phi_{z_1}(d) \log \frac{\Phi_{z_1}(d)}{\Phi_{z_2}(d)}
\]

In the case of study, the expectation is taken using the probabilities of each domain \(d \in D\). \(\Phi_{z_1}\) and \(\Phi_{z_2}\) are the domain probabilities for environments \(z_1\) and \(z_2\) \((\Phi_{z_1}(d) = P(d|z_1)\) and \(\Phi_{z_2}(d) = P(d|z_2))\). Therefore, environments \(z\) quite different from all the others will have a large value of \(D_{KL}(z) = \sum_{z_i} D_{KL}(\Phi_{z} \parallel \Phi_{z_i})\), that I call KL divergence for environment \(z\).

I further compute the mean expected number of visits for each environment, namely the popularity of the environments \(z\), computed as \(V(d) \cdot P(d|z)\), where \(V(d)\) is the number of visits towards domain \(d\) in the original dataset.

I plot \(D_{KL}(z)\) versus \(V(d) \cdot P(d|z)\) for each \(z\). Figure 5.7 shows results for: (a) the whole Campus, (b) Electronic, and (c) Architecture departments. Each environment is labeled with a different number for ease of visualization. We see in all cases a clear trend: the most popular environments are also the one that are less diverse from each other. These environments are generic, and thus less interesting to inspect. On the other hand, environments with large KL divergence usually have a low number of expected visits. These environments are the most peculiar ones. For example, consider environment 2 in Campus traces. Analyzing its most popular domains, it appears related to users/devices that often go to visit sony.com and sony.it, possibly because they hosts home-pages automatically visited by software installed on those users’ devices. Other examples of peculiar environments include domains related to torrents and movie streaming portals, and to popular bank accounts. This method of analysis offers the analyst an automatic way to highlight interesting environments.

### 5.4.3 Transitions in communities

In the previous subsections I showed how interests are different in communities of people. I now show how communities switch among environments. In figure 5.8 I depict the heat-maps of the transition probability matrix \(P(z_1|z_2)\) from environment \(z_1\) to another environment \(z_2\), in logarithmic scale. This matrix can be easily computed from the model parameters. By construction, elements of a row has to sum to 1, while the sum on the element of a column represents the probability of landing to that particular environment. I sort the environments by such (reversed) sum
of probabilities. First, it can be seen how at each step it is more likely to stay within the same environment (see the dark diagonal). However, at least for my datasets, other transitions are usually towards a single popular environment that contains generic web-sites (on the foremost left in figure 5.8(a)). On the other hand, transitions on Architecture and Electronic departments are more spread towards many topics, some of which are specific of each department. Among all the possible 870 transitions, only 86 for the Campus are above the random probability, while this number increases to 129 for Electronic and 146 for Architecture department. This means that inside the departments people are much more homogeneous and tends to visit more evenly the peculiar environments. Even if very different between each others, these environments are in general more common within the same community of people.

5.5 Conclusions and future work

In this chapter, I described how to create from passive traces meaningful trajectories of users on the web, and then how to represent them in a succinct and interpretable manner. I showed that the model allows me to automatically and easily inspect the interests of users and communities, and to highlights transitions and diversity is such clusters of domains. I plan to extend my work in order to: (i) study temporal evolution of trajectories and environments, (ii) apply my methodology to much bigger datasets coming from ISPs, and (iii) propose personalized recommendation systems for users browsing.
Chapter 6

User behaviour in online advertising: modeling and optimization of temporal contextual ads

While in the previous chapters I have shown how to extract and model the user behaviour on the web from passive traces, in this chapter I focus on the influence of online advertisement on the users. I introduce a user behavioural model that specifically captures the impact on performance of the history of impressions shown to each user. The proposed model, validated by passive traces of real advertising systems, allows optimizing the revenue under different scenarios.

The content of this chapter is currently under review [69]. Proofs presented in sections 6.3.4, 6.4 and 6.5 were primarily work of my collaborators.

6.1 Introduction

In recent years we have seen a proliferation of online platforms offering new types of (usually free) services to increasing populations of users. Beyond traditional services such as web-pages, search engines, blogs, several companies have built extraordinary profits out of popular activities such as social networking, image/video sharing, instant messaging, cloud storage, and many others. Even when services are offered entirely for free, huge profits can be obtained for mainly two reasons: i) big
data analytics (i.e., the collection, analysis and exploitation of information acquired by tracking the users’ activity) and ii) ad sales (i.e., the insertion of advertisements within the content displayed to users). The above two sources of profit are naturally intertwined: data analytics is typically used to build user profiles which are then exploited for targeted advertisements.

The success of an online advertising campaign is traditionally measured by what is referred to as click-through-rate (CTR), which is defined as the number of times a click is made on the ads belonging to the campaign, divided by the number of times these ads are shown (impressions) [70]. I mention here that simply counting the number of clicks has been found to be a biased/incomplete metric, also prone to fraud, while richer user activities (such as different forms of interaction with media, site sign-ups, or, eventually, actual conversions to purchase) should be considered to better capture the user response. However, all possible actions that users can perform have been invariably measured as ratios (or percentages), i.e., as number of actions divided by number of impressions. For this reason, it can be said that the current practice to measure the effectiveness of an online ad campaign is a generic ‘action-through-rate’.

When we try to maximize the revenue derived from an ad campaign, we cannot just consider the action-through-rate alone, except for the unrealistic case in which CTR is not affected by the number of impressions shown to the users. In reality, the history of impressions shown to each specific user (i.e., their number and temporal spacing) can have a profound impact on his/her likelihood to perform a valuable action on them, as it has often been recognized [71–75].

Therefore, a careful optimization of the temporal spacing by which impressions are shown to the user is actually needed to maximize the success of an ad campaign in terms of number of clicks or richer user actions.

The above considerations suggested to adopt a radically different point of view, which brings the history of impressions back into the problem by taking into account the temporal dynamics of the advertising system and the user. In particular, I look at the system in terms of online streams, as illustrated in figure 6.1. An Ad Server (centralized or distributed, i.e., an Ad network) receives from the Publisher a stream of available slots (i.e., portions of the user’s navigation experience where ads can be inserted), while it receives from the Advertiser a stream of ads that can potentially fill
in those slots. A stream of impressions (in general, a subsets of available slots which have been matched with ads) is shown to each given user, who might decide to click on them, or, more in general, perform valuable actions on them, generating a stream of clicks or actions. The rate of such stream, that I call CTI (Click Through Intensity) or more in general, ATI (Action Through Intensity) and expressed in number of clicks per time unit, is the novel performance metric that I seek to maximize. Note that this metric is directly proportional to the actual revenue that can be obtained from a campaign lasting for a given time. The intelligence of the system resides in the Ad Server, which decides which impressions to deliver to each specific user, and at which time instants, usually guided by analytics collected on the user itself.

The reason why I consider detailed temporal dynamics (by closely looking at the arrival processes of slots/ads/impressions) is the fact that I incorporate a model of user behaviour, which provides the likelihood that the user will perform a valuable action on a particular impression shown at a given time. Intuitively, a user overwhelmed with impressions arriving too close in time is less likely to perform actions on them (e.g., because he/she get annoyed by the ads).

The contributions of my work can be summarized as follows.

1. I propose a new metric, the Click-Through-Intensity, instead of the traditional Click-Through-Rate, in order to effectively quantify the success of an advertising campaign;

2. I introduce a stochastic framework that allows optimizing the proposed metric considering the temporal dynamics of an online system coupled with the dynamics of user behaviour;
3. I identify different regimes of the proposed system and characterize their performance by analysis and simulation;

4. I provide, for the first time to the best of my knowledge, an analytical approach to optimize the frequency capping of an ad campaign;

5. I validate my approach using traces taken from real advertising systems.

6.2 Related work

Automated and semi-automated means of optimizing online ad placement, targeting, bid prices, have being subject of extensive investigation since the advent of the web, for obvious reasons related to their huge monetary value. With massive deployment, click-through-ratios for banner ads have fallen over time to values as low as 0.2-0.3 percent.\(^1\) Although very aggressive ways to display ads to the users have been commonly adopted, most publishers have realized that the success of a campaign depends crucially also on the quality of user engagement. In particular, overly intrusive ads are often completely avoided by viewers, and tend to have a very detrimental effect on user experience [76, 77].

Nowadays, behavioural (or contextual) targeting has emerged as one of the main techniques to increase the efficiency and profits of digital advertisement, thanks to sophisticated tracking and profiling platforms. In [78] authors analyze the economic implications of behavioural targeting, showing that in some circumstances the revenue for the publisher can double by using this technique, though the gain depends on many factors such as the nature of competition between large and small publishers. The competitive interaction between content producers (e.g., advertisers) over a social network, and the actions that producers can take in order to increase their content visibility, has been studied through the lens of game theory in [79].

Several empirical studies have investigated the benefits of behavioural targeting, focusing on traditional click-through-ratios (e.g., [80, 81]). Further, it has been recognized that, beyond conventional accuracy metrics, new user-centric approaches taking into account human factors can significantly improve the user experience. The

\(^1\) Andrew Stern, 8 ways to improve your click-through rate, 
role of human factors is well understood in marketing research, where there exists large consensus that purchase events are driven by two main factors, a "habituation", or inertia, factor and a "boredom", or variety-seeking factor, that comes into play when consumers get "satiated" or "bored" with the same product over time. Such tendencies can coexist within the same consumer, evolving over time [82].

Similar human factors have been considered in the related field of recommendation systems based on collaborative filtering [83], where the goal is to improve the prediction of future ratings for a user-item pair. In addition to accuracy metrics, several authors have suggested various ways to combine other metrics such as diversity, coverage, and serendipity [84–87]. There have been also several attempts to incorporate temporal dynamics of user behaviour in standard techniques developed for recommendation systems. In particular, time-decaying weight functions are usually introduced to capture the fact that more recent data better reflect a user’s current preference [88, 89]. However, tracking the temporal dynamics of customer preferences is generally recognized as a challenging problem [90], due to the existence of several possible interest drift patterns [75].

The majority of existing models ignore the fact that users may get annoyed/bored of recommendations, despite their past interactions. One exception is the work in [91], where authors show that user’s temporal consumption of familiar items is driven by boredom. They propose a semi-Markov model with two latent psychological states, sensitization and boredom, to characterize the user revisit times to an item. The authors of [74] also introduce boredom to explain the cyclic pattern of individual choices as well as social trends. They propose a model in which boredom is proportional to the total accumulated memory for an item, where each consumption of the item adds a term that drops geometrically over time. The analysis in [92] accounts for the temporal evolution of user interests through attraction or aversion towards past suggestions. In [73] authors propose a model of user response to an ad campaign as a function of both the interest match and the past exposure.

To contain the negative effects of user boredom/annoyance, many existing ad serving technologies already offer to advertisers a configurable option called frequency capping. For example, the popular platform Google Adwords allows setting a limit to the number of impressions an individual user will see per day, per week, or
per month.\textsuperscript{2} Although frequency capping is slowly becoming a standard practice, there are no well established methodologies to set the threshold, and most advertisers resort to trial-and-error or rough guidelines (recommended values are in the range of 3 views/visitor/24 hours). To the best of my knowledge, I am the first to theoretically investigate the problem of optimizing the frequency capping of an ad campaign using a behavioural model.

6.3 System model

I introduce here the assumptions on the various components of the advertising system illustrated in figure 6.1. First, I describe how a user is affected by received impressions (section 6.3.1). Second, I characterize how likely the user will click (or, more in general, perform the desired action) on a given impression (section 6.3.2). Then, I explain the different scenarios that arise depending on how impressions can be delivered to the user by the Ad Server (section 6.3.3). After that, I will introduce the stateless thinning strategy (SLT), which clearly demonstrates the need to optimize CTI, and not CTR, to maximize the revenue of an ad campaign (section 6.3.4). The SLT strategy will also be used as baseline while evaluating the performance of my approach.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{user_excitation.png}
\caption{Example of evolution of user excitation $U(t)$.}
\end{figure}

6.3.1 Model of user excitation

I associate to each user a time-varying, non-negative real value \( U(t) \) that I call user excitation. The user excitation keeps track of the cumulative effect of the impressions shown to the user, assuming that the impact (deterministic or random) of each impression decays over time.

Figure 6.2 shows an example of the temporal evolution of \( U(t) \) in the case of a user receiving a sequence of four impressions displayed at times \( t_i, i = 1 \ldots 4 \). The excitation, which is initially set to zero, is incremented by \( L_i \) when impression \( i \) arrives. In the simplest case, \( L_i \) is just a constant, but for greater generality I allow it to be a random variable accounting for various effects related to the user behaviour, the environment, etc. We can assume, without lack of generality, that user excitations have mean equal to 1, i.e., \( E[L_i] = 1 \). I will consider two different cases:

- **Perfect excitation information.** I assume here that the Ad Server, in addition to times \( t_i \), has also knowledge of values \( L_i \). This is an analytically tractable ideal case, in which we can fully characterize the optimal strategy and derive upper bounds to the system performance.

- **Partial excitation information.** In this more realistic case, the Ad Server, in addition to times \( t_i \), has inferred a probability distribution over \( L_i \), thanks, for example, to detailed tracking of the interaction between the user and the ad (e.g., by taking into account the time spent watching a video associated to the ad, or by considering how far the user has proceeded along the way leading to the actual purchase of the sponsored item).

For both cases above, although we could consider the general case in which each \( L_i \) has its own distribution, I will assume, for simplicity, that \( L_i \) are i.i.d. positive random variables with the same distribution. I further assume that the distribution of \( L_i \) has all polynomial moments finite.

As usually done in many physical systems, and supported by the Ebbinghaus’s model [93], the impact on the user of each stimulus (impression) is assumed to decay exponentially over time, with parameter \( \alpha \), representing in this context the physiological process of ‘forgetting’. Parameter \( \alpha \) can be inferred for the specific user, or just estimated from the average behaviour of a larger set of users. We will
assume, for now, that the exact value of $\alpha$ is known to the system. In the performance evaluation section, we will consider the impact of an erroneous estimate of $\alpha$.

Note that, under perfect information on $L_i$ and $\alpha$, the system has full knowledge of the instantaneous user excitation $U(t)$. By denoting with $t$ the current time instant and with $t_i$ the time instant at which the user has received the $i$-th previous impression, the current value of user excitation is given by:

$$U(t) = \sum_{t_i < t} L_i e^{-\alpha(t-t_i)}.$$  \hspace{1cm} (6.1)

I remark that we will assume $U(t)$ to be left continuous.

### 6.3.2 Model of user response

The user excitation introduced in previous sections is instrumental to another essential component of the model, which evaluates the probability that the user performs the desired action (just a click, or a more complex interaction, or an actual purchase) on a given impression. My main assumption here is that the user will perform the desired action on the impression displayed at time $t_i$ with a probability $P_{a_i}(U(t_i) + L_i)$ that depends on the current value of user excitation (including the impact of the impression just received, i.e., value $L_i$). I will refer to $P_{a_i}(x) : \mathbb{R}^+ \rightarrow [0,1]$ as the user response function, and again assume that this function is known to the system, having been inferred for the specific user or estimated across a larger set of users.

I now provide qualitative examples of user response function $P_{a_i}$ derived from experimental data. This requires to first introduce the traces of real advertising systems used in my work.

The Avazu dataset, publicly available over the Kaggle platform,\(^3\) reports the click/no-click actions performed by 9 million mobile users on on-line ads, over 10 days. To limit the impact of noisy data/outliers, I decided to restrict my attention to a core of users having the following characteristics: i) they have been exposed to a number of ads between 24 and 2400 during the 10 days of the trace, with a maximum of 60 ads per hour; ii) they have performed a click-through rate between 0.05 and 0.25. While the publicly available trace contains all click events, no-click

Figure 6.3 $P_a$ obtained from (a) Avazu and (b) Outbrain datasets.
events have been subsampled by Kaggle to limit the trace size, which explains why
the observed click-through rates are significantly larger than what usually reported in
the literature (CTR below 0.01). This filtered dataset contains 2 million impressions
and 21k users.

Assuming, in the absence of further information, that $L_i$ are deterministic (equal
to 1), I computed the evolution of the shot-noise process $U(t)$ for each user using the
impression arrival times reported in the trace, trying different values of $\alpha$. Since the
time granularity of the dataset is in hours, I considered ads in the same hour to be
equally spaced over the 1-hour interval (preserving the order in which they appear in
the trace). For each value of $\alpha$, I have then evaluated the corresponding empirical
user response function, using the click/no-click information in the trace. To avoid
transient effects, I discarded for each user the first $2/\alpha$ hours of the reconstructed
$U(t)$ process.

Figure 6.3(a) reports the empirical $P_\alpha$ for all considered users, in the case of
$\alpha = 0.3$. The dashed line in the plot shows the best least-square fitting of the
experimental data by an exponential function, that I will use later on (section 6.4.5).
Results in figure 6.3(a) reveal a strong correlation between the user response and
the user excitation, suggesting that the methodology can be effectively employed
to optimize the system. I obtained similar results for other values of $\alpha$ around 0.3,
whereas correlation tends to vanish for much smaller (0.01) or much larger (10)
values.

I repeated the same experiment with the Outbrain dataset, also available on
the Kaggle platform.\footnote{Outbrain dataset, \url{https://www.kaggle.com/c/outbrain-click-prediction}, accessed January 2018.} The trace contains click/no-click information on roughly
87 million impressions over a period of 14 days (including accurate timestamps),
though again no-clicks have been significantly subsampled to reduce the trace size.
I restricted my attention to the 10 000 most active users, accounting for $\approx 500000$
impressions. Figure 6.3(b) shows the empirical $P_\alpha$ obtained again for $\alpha = 0.3$.
Interestingly, results are similar to those coming from the Avazu dataset. They
suggest that, in both datasets, user response is mainly affected by a boredom effect,
producing a monotonically decreasing user response function.

In marketing research, it is also common to consider a non-monotonous user
response function with a single peak (i.e., a function initially increasing up to a
maximum value, and then decreasing), to jointly account for habituation (mere-exposure effect) and boredom. An “inverted-U” shaped function has been justified by Berlyne’s theory of exploratory behaviour [94], who first studied the relationship between attractiveness of a stimulus and its familiarity (number of repetitions). Since such a theory has been largely adopted in marketing research, in the evaluation I will also consider a non-monotonous $P_a$, in addition to a monotonically decreasing function like the simple exponential fitting shown in figures 6.3(a) and 6.3(b). I remark, however, that in the theoretical analysis I do not pose particular restrictions to function $P_a$ (e.g., it does not even have to be continuous). However, to avoid trivialities, I always assume that $P_a(x) \to 0$ for $x \to \infty$.

At last, I emphasize that, although the way in which I have fitted the model to the Avazu and Outbrain datasets is largely arbitrary, due to intrinsic limitations of the information available in the traces, the main point that I want to make is that user response and user excitation (as defined in the model) are indeed significantly correlated, at least over appropriate time scales. If $P_a$ were independent, or very weakly correlated with $U$ (e.g., a flat function), there would be no room for any optimization (i.e., no reason to introduce any frequency capping): the best strategy, in this case, would be to just overwhelm the users with ads, exploiting all possible impression opportunities. However, this is clearly not the case, as confirmed by the traces.

### 6.3.3 Timeliness of ads delivery

In my analysis, I focus on an specific ad campaign performed by an Advertiser through a particular Publisher. The problem of maximizing the revenue of an ad campaign addressed to many potential users naturally splits into orthogonal subproblems aimed at maximizing the revenue coming from single users. As already mentioned, nowadays sophisticated tracking and profiling technologies permit designing advertising strategies specifically tailored to individuals (usually identified by cookies or explicit logins). Therefore, I will focus on just one user, whose profile (interests, navigation habits, response history to past advertisements) is known to the system. It is intended here that, in the absence of information about the user (e.g., a new visitor or a new service subscriber), the system will initially use average characteristics of its known users.
I will further assume that candidate ads to be sent to the user are qualitatively similar, i.e., they can be considered to be equally interesting to the user according to its current profile information. Note that the Ad Server (see figure 6.1) has to match candidate ads of the Advertiser with available slots of the Publisher, and thus operates in a way similar to a leaky bucket, a rate control algorithm well known in data networks. In the following, I will refer to a matched pair (ad+slot) as an impression opportunity.

For such model, I will analyze the following two extreme scenarios:

- **Arbitrary delivery.** Here, the system can send impressions to the user at arbitrary time instants. Although this assumption is unrealistic, it provides an upper bound to the system performance. Moreover, this case can be considered as a good approximation of systems in which there is abundance of impression opportunities, i.e., when we jointly have (see figure 6.1): i) abundance of slots, meaning that the user is online often enough (as compared to the time-scale of the optimal ad delivery rate) to be considered permanently exposed to the advertising system; ii) abundance of ads, meaning that new ads arrive at sufficiently large rate, and they are delay-tolerant (i.e., they can be significantly delayed after they are first generated, maintaining equal interest to the user, or even indefinitely repeated). Thus, the advertising system has the freedom to arbitrarily choose the temporal spacing with which impressions are displayed to the user (I will show that, under any reasonable user response function, the optimal strategy is to cap the frequency of ad delivery).

- **Real-time delivery.** Here I assume that, either because of scarcity of slots, or because of scarcity of ads, the system is forced to select the impressions to send to the user from online stream of finite rate \( \lambda \). In particular, I will consider the case in which impression opportunities become available to the system according to a Poisson process of rate \( \lambda \). I will further assume that the system has to make an instantaneous decision whether to send each impression to the user or not, at the same time at which the pair becomes available. This might be due to the fact that ads cannot be delayed (i.e., they soon expire after they are generated).
I will also explore by simulation the intermediate case in which impression opportunities arrive at finite rate $\lambda$, but remain valid for some (known) time (deterministic or random), after which they expire and cannot be sent to the user any more.

### 6.3.4 The SLT strategy and the need of a new approach

As mentioned, the revenue of an ad-campaign is directly proportional to the Click-Through-Intensity, thus the objective will be to maximize CTI, for the various scenarios introduced above.

Before diving into the analysis, it is instructive to see what happens under the following hypotheses: i) impression opportunities arrive according to a Poisson process of rate $\lambda$; ii) the Ad Server just delivers the impressions to the user as soon as they arrive.

For the purpose of numerical illustration, Figure 6.4 reports CTI as function of the arrival rate $\lambda$, assuming deterministic $L_i = 1$, $\alpha = 1$, and the monotonically decreasing response function $P_a = 0.1e^{-u}$. Figure 6.4 clearly shows that there exists an optimal ad-delivery rate maximizing CTI.

Based on the above result, one could devise a very simple strategy, called State-Less Thinning (SLT), which simply performs a blind thinning of the arriving stream of impression opportunities, by actually sending each opportunity to the user with fixed, independent probability $p$.

**Proposition 6.3.1.** The optimal thinning probability $p$ of a SLT strategy can be analytically computed under a Poisson arrival process of impressions, assuming knowledge of $P_a$ and distribution of $L_i$.

**Proof.** Let $M_U(\theta) = \mathbb{E}_U [e^{\theta U(t_n)}]$ the moment generating function associated to the stationary distribution of $U(t_n)$. Since $\{t_n\}_n$ is a Poisson process, we have $M_U(\theta) = \mathbb{E}_U [e^{\theta U(0)}]$, since by PASTA [95] we can consider any arbitrary time instant (namely, 0). Now, by construction:

$$U(0) = \sum_{t_n \in (-\infty, 0)} L_n e^{\alpha t_n}$$
We also define the truncated version:

\[ U_T(0) = \sum_{t_n \in [-T,0)} L_n e^{\alpha t_n} \]

The moment generating function of \( U_T(0) \), denoted by \( M_U^T \), can be easily obtained by conditioning on the number of points of \( \{t_n\} \) that fall in \([-T,0)\). Therefore, we can exploit the following property of Poisson processes: conditionally over their number, the non-ordered points, \( Z_i \), of a Poisson process falling over a finite interval are uniformly and independently distributed. It follows that:

\[
M_U^T(\theta \mid m) = \mathbb{E} \left[ e^{\theta \sum_{i=1}^{m} L_i e^{\alpha Z_i}} \right] = \mathbb{E} \left[ \prod_{i=1}^{m} e^{\theta L_i e^{\alpha Z_i}} \right] = \left( \mathbb{E} e^{\theta L_1 e^{\alpha Z_1}} \right)^m
\]

with:

\[
\mathbb{E} \left[ e^{\theta L_1 e^{\alpha Z_1}} \right] = \frac{1}{T} \int_{-T}^{0} \int_I e^{\theta L e^{\alpha Z}} \partial F_L(l) \partial z
\]

Therefore,

\[
M_U^T(\theta \mid m) = \left( \frac{1}{T} \int_{-T}^{0} \int_I e^{\theta L e^{\alpha Z}} \partial F_L(l) \partial z \right)^m
\]

and, unconditioning:

\[
M_U^T(\theta) = \sum_{m=0}^{\infty} \left( \frac{1}{T} \int_{-T}^{0} \int_I e^{\theta L e^{\alpha Z}} \partial F_L(l) \partial z \right)^m \frac{(\lambda p T)^m}{m!} e^{-\lambda p T}
\]

\[
= \exp \left( -\lambda p T + \lambda p \int_{-T}^{0} \int_I e^{\theta L e^{\alpha Z}} \partial F_L(l) \partial z \right)
\]

\[
= \exp \left( -\lambda p \int_{-T}^{0} \int_I (1 - e^{\theta L e^{\alpha Z}}) \partial F_L(l) \partial z \right)
\]

Then for \( T \to \infty \), we get:

\[
M_U(\theta) = \exp \left( -\lambda p \int_{-\infty}^{0} \int_I (1 - e^{\theta L e^{\alpha Z}}) \partial F_L(l) \partial z \right)
\]

\[
= \exp \left( -\lambda p \int_{0}^{\infty} \int_I (1 - e^{\theta L e^{\alpha Z}}) \partial F_L(l) \partial z \right)
\]

(6.2)

Observe that, if we assume \( L \) to have a compact support, then for \( z \to \infty \), \( le^{-\alpha z} \to 0 \), hence, \( \exp (\theta le^{-\alpha z}) = 1 + \theta le^{-\alpha z} + o(e^{-\alpha z}) \). In other words, the integrand function decays to 0 exponentially fast as \( z \to \infty \).
Due to the complexity of (6.2), we aim at deriving a simpler expression for $M_U(\theta)$. To this end, for every $T > 0$, we write:

$$
\int_0^T \int_l \left( 1 - e^{\theta l e^{-\alpha z}} \right) \partial F_L(l) \partial z = \\
= \int_0^T \int_l \left( 1 - \sum_{k=0}^{\infty} \frac{(\theta l)^k}{k!} e^{-k\alpha z} \right) \partial F_L(l) \partial z \\
= \sum_{k=1}^{\infty} \frac{\theta^k}{k!} \int_0^T e^{-k\alpha z} \partial z \int_l l^k \partial F_L(l) \\
= \sum_{k=1}^{\infty} \frac{\theta^k}{k!} \frac{1 - e^{-k\alpha T}}{k\alpha} \mathbb{E}_L[L^k]
$$

Thus,

$$
\int_0^{\infty} \int_l \left( 1 - e^{l e^{-\alpha z}} \right) \partial F_L(l) \partial z = \\
= \lim_{T \to \infty} \sum_{k=1}^{\infty} \frac{\theta^k}{k!} \frac{1 - e^{-k\alpha T}}{k\alpha} \mathbb{E}_L[L^k] \\
= \sum_{k=1}^{\infty} \frac{\theta^k}{k!} \frac{\mathbb{E}_L[L^k]}{k\alpha}
$$

(6.3)

We therefore obtain:

$$
M_U(\theta) = \exp \left( -\lambda p \sum_{k=1}^{\infty} \frac{\theta^k \mathbb{E}_L[L^k]}{k\alpha} \right)
$$

(6.4)

From (6.4), it is then straightforward to derive the stationary distribution $\pi_U(u)$, which will depend on $p$. It follows that the optimal policy, under the assumption that the filtered arrival process is still Poisson, can be found by replacing the expression of $\pi_U(u)$ in the revenue and by optimizing the latter with respect to $p$.

Note that, assuming $\lambda$ large enough, as we vary $p$ the above SLT strategy would precisely achieve the performance shown in figure 6.4 (regarding the $x$ as the rate of thinned impressions). Therefore, an optimal choice of $p$ (possibly by trial and error) would essentially allow achieving the maximal value of CTI appearing in figure 6.4.

I argue that the simple SLT strategy, if adopted, would dramatically improve the revenue of most online campaigns, which simply overwhelm users with uncapped
impressions (thus operating on the rightmost portion of the curve shown in figure 6.4). In the following, as one of the main contributions of my work, I will show that, in general, the SLT strategy is not optimal, meaning that more sophisticated filtering strategies can achieve superior CTI, under both arbitrary and real-time delivery scenarios. Although the additional gains that my strategies can achieve with respect to SLT are not impressive (around 10%-30%), I will be able to show their optimality in the corresponding scenarios.

I conclude this section with a final interesting observation. In the above numerical example, what would happen if we tried to maximize the traditional Click-Through-Rate (CTR), instead of CTI? One can easily see that, under any strictly decreasing user-response, CTR is maximized under vanishing ad-delivery rate, i.e., by waiting until the user excitation $U$ becomes very small, so that the next opportunity will be accepted with the highest probability. Clearly, so doing one would achieve vanishing CTI, hence vanishing revenue, confirming that CTR is not the right metric to consider, in order to maximize the revenue of an ad campaign.

6.4 Analysis under perfect excitation information

I start with the case in which the system has full knowledge of the user excitation $U(t)$. 
Let \( \{t_i\}_i \) be the sequence of times at which the user is exposed to impression opportunities (i.e., right before the excitation increment due to impression delivery). We observe that \( U(t_i) \) is uniquely determined by the triplet \( (U(t_{i-1}), L_{i-1}, \tau_i) \), with \( \tau_i = t_i - t_{i-1} \). In other words, assuming \( U(t_{i-1}) \) to be given, \( U(t_i) \) is conditionally independent of \( \{U(t_j)\}_j \) for \( j < i - 1 \), whenever \( \tau_i \) is conditionally independent of \( \{\tau_j\}_{j<i}, \{U(t_j)\}_{j<i-1}, \{L_j\}_{j<i-1} \).

We can thus restrict ourselves to study Markovian policies (i.e., consider policies according to which \( \{\tau_i\}_i \) satisfies previous properties), and prove that \( \{U(t_i)\}_i \) forms an ergodic Markov process over \( \mathbb{R}^+ \) (or a compact subset of \( \mathbb{R}^+ \)) under some additional weak assumptions on the distribution of \( L_i, \tau_i \).

**Proposition 6.4.1.** Assume that: (i) \( L_i \), which represents the increment in the user excitation upon the delivery of an impression opportunity, exhibits finite polynomial moments, and (ii) \( \tau_i \) exhibits finite polynomial moments and \( \tau_i > \delta \) with probability one, for some \( \delta > 0 \), whenever \( U(t_{i-1}) \) is sufficiently large. Then \( \{U(t_i)\}_i \) represents an ergodic Markov process.

**Proof.** A proof of the ergodicity of \( \{U(t_i)\}_i \) can be given with standard drift arguments. Indeed, \( \mathbb{E}[U(t_i) \mid U(t_{i-1})] = \mathbb{E}[(U(t_{i-1}) + L_{i-1})e^{-\alpha\tau_i}] < (\mathbb{E}[U(t_{i-1})] + \mathbb{E}[L_{i-1}])e^{-\alpha\delta} = \mathbb{E}[U(t_{i-1})] - \mathbb{E}[U(t_{i-1})](1 - e^{-\alpha\delta}) + \mathbb{E}[L_{i-1}]e^{-\alpha\delta} < \mathbb{E}[U(t_{i-1})] - 1 \) whenever \( \mathbb{E}[U(t_{i-1})] \) is sufficiently large. \( \square \)

Being the process ergodic, we can define \( \pi_U(u) \) as the unique stationary distribution of \( \{U(t_i)\}_i \) over a properly defined support. In the following, we study the \( \{U(t_i)\}_i \) process under stationary conditions and denote by \( U \) the random variable representing the user excitation at the generic time \( t_i \). We also denote by \( L_i \) the generic \( L_i \).

Given the above observations and assumptions, below we derive the optimal Markovian policy under both arbitrary delivery (section 6.4.1) and real-time delivery (section 6.4.2) scenarios.

### 6.4.1 Arbitrary delivery

Recall that in this scenario the Ad Server can deliver impression opportunities to the user at arbitrary time instants. I will show that in this case the policy that maximizes
CTI is the one that sends a new impression opportunity to the user whenever the excitation $U(t)$ goes back to a fixed, optimal value $\theta_0$, which depends on the response function $P_a$ and the distribution of $L_i$.

I start by proving the following proposition, which expresses CTI as the ratio between the average revenue obtained by an impression and the average time between two consecutive opportunities shown to the user.

**Proposition 6.4.2.** The CTI achieved by a Markovian policy with stationary distribution $\pi_U(u)$ is given by:

$$\text{CTI} = \frac{\alpha \mathbb{E}_U \mathbb{E}_L [P_a(U + L)]}{\mathbb{E}_U \mathbb{E}_L \left[ \log \left(1 + \frac{L}{u}\right)\right]} = \frac{\alpha \int_{\tilde{u}}^u P_a(u + l) \partial F_L(l) \partial \pi_U(u)}{\int_{\tilde{u}}^u \log \left(1 + \frac{l}{u}\right) \partial F_L(l) \partial \pi_U(u)}.$$  \hspace{1cm} (6.5)

**Proof.** The above result is obtained by applying renewal theory. The Ad Server receives a revenue whenever the user is exposed to an impression opportunity and the user takes action. Note that, under an ergodic Markovian policy, the evolution of the user excitation is Markovian. Furthermore, it can be seen that, under any ergodic policy with stationary distribution $\pi_U(u)$, the evolution of the user excitation over continuous time forms a regenerative process. This because choosing an arbitrary $\tilde{u}$ such that $0 < \pi_U(\tilde{u}) < 1$ and defining with $\{\tilde{\sigma}_i\}_i$ the sequence of times at which $U(t) = \tilde{u}$, we can show that $\{\tilde{\sigma}_i\}_i$ forms a non defective sequence of stopping times for the continuous time process $U(t)$ (i.e., $\mathbb{E}[\tilde{\sigma}_i - \tilde{\sigma}_{i-1}] < \infty$). Indeed, I define $S_{\tilde{u}}$ as the set of states in correspondence of which $U(t) \leq \tilde{u}$ and $S_{\tilde{u}}$ as the complementary set corresponding to values $U(t) > \tilde{u}$. Then we can observe that: i) by ergodicity, the sequence of times $\{\sigma_i\}_i$, at which the system enters a state in $S_{\tilde{u}}$ from a state $S_{\tilde{u}}$ forms a non defective sequence; ii) by construction, $\{\sigma_i\}_i \subseteq \{\tilde{\sigma}_i\}_i$, therefore $\{\sigma_i\}_i$ is non defective.

It follows that we can apply standard Reward-Renewal results [96], according to which the average revenue if given by

$$\mathcal{R} = \frac{\mathbb{E}[P_a(U + L)]}{\mathbb{E}[^\tau]}$$  \hspace{1cm} (6.6)

where $\tau$ is the intertime between two generic impression opportunities to which the user is exposed. In order to derive the denominator, we express the user excitation at time instant $t_{i+1}$ (time at which the user is exposed to impression opportunity $i + 1$), as a function of the user excitation at time instant $t_i$ (time at which the user has been
exposed to opportunity $i$):

$$U(t_{i+1}) = (U(t_i) + L_i)e^{-\alpha \tau_{i+1}}$$  \hfill (6.7)

where $\tau_{i+1} = t_{i+1} - t_i$. Thus, we get:

$$\tau_{i+1} = \frac{1}{\alpha} \log \frac{U(t_i) + L_i}{U(t_{i+1})}$$  \hfill (6.8)

from which we obtain:

$$\mathbb{E}[\tau] = \frac{1}{\alpha} \mathbb{E}_L \mathbb{E}_U \left[ \log (U(t_i) + L_i) - \log U(t_{i+1}) \right]$$

$$= \frac{1}{\alpha} \int\int_{l,u} \left[ \log (u+1) - \log u \right] \partial \pi_U(u) \partial F_L(l)$$

$$= \frac{1}{\alpha} \int\int_{l,u} \log \left( 1 + \frac{l}{u} \right) \partial \pi_U(u) \partial F_L(l).$$  \hfill (6.9)

By replacing (6.9) into (6.6), we get the thesis. \hfill $\square$

Looking at (6.5), it can be seen that the CTI depends on the selected policy only through the stationary distribution $\pi_U(u)$. Therefore, all the Markovian policies with the same stationary distribution lead to the same performance. In particular, we are interested in the optimal policy associated to

$$\pi^*_U(u) = \arg\max_{\pi_U(u)} \frac{\mathbb{E}_U \mathbb{E}_L [P_d(U + L)]}{\mathbb{E}_U \mathbb{E}_L \left[ \log \left( 1 + \frac{L}{\theta} \right) \right]}$$

where for now we assume that a maximum exists over $\pi_U(u)$ (its existence is shown below).

Next, within the class of Markovian policies, we define a subclass of policies enforcing $U(t_i) = \theta, \forall i$, i.e., policies that expose the user to a new impression opportunity whenever the excitation goes back to a given threshold $\theta$. In the following, we will generally refer to such policies as threshold-based.

For threshold-based policies, the expression of CTI reduces to:

$$\text{CTI} = \frac{\alpha \int_1 P_d(\theta + l) \partial F_L(l)}{\int_1 \log \left( 1 + \frac{l}{\theta} \right) \partial F_L(l)}. $$
Then we define:

$$\theta_0 = \arg \max_{\theta} \frac{\int P_a(\theta + l) \partial F_L(l)}{\int \log \left(1 + \frac{l}{\theta} \right) \partial F_L(l)}.$$  \hspace{1cm} (6.10)

where $\theta_0 < \infty$ as a consequence of the assumptions on $P_a$ and $\{\tau_i\}$. Below, we show that the threshold-based policy with $\theta_0$ is optimal among all Markovian policies.

**Theorem 6.4.3.** Given $\theta_0$ as defined in (6.10), the optimal among all Markovian policies is the threshold-based policy using $\theta_0$.

**Proof.** We denote with $f(\theta) = \int P_a(\theta + l) \partial F_L(l)$ and $g(\theta) = \int \log \left(1 + \frac{l}{\theta} \right) \partial F_L(l)$. Note that clearly $f(\theta)$ and $g(\theta)$ are non-negative functions, with $g(\theta) > 0 \forall \theta > 0$.

It can be shown that:

$$\frac{\int f(\theta) \partial \pi_U(\theta)}{\int g(\theta) \partial \pi_U(\theta)} = \frac{\int [f(\theta) - f(\theta_0)] \partial \pi_U(\theta) + f(\theta_0)}{\int g(\theta) - g(\theta_0) \partial \pi_U(\theta) + g(\theta_0)} = \frac{f(\theta_0)}{g(\theta_0)} \left( \frac{1 + \int \left( \frac{f(\theta)}{f(\theta_0)} - 1 \right) \partial \pi_U(\theta)}{1 + \int \left( \frac{g(\theta)}{g(\theta_0)} - 1 \right) \partial \pi_U(\theta)} \right).$$

Now we get the assertion by proving that

$$1 + \int \left( \frac{f(\theta)}{f(\theta_0)} - 1 \right) \partial \pi_U(\theta) \leq 1 + \beta \leq 1.$$

Of course the assertion is trivially true if $\beta > \alpha$, i.e., $\alpha - \beta \leq 0$. The latter expression holds since by construction: $\alpha - \beta = \int \left( \frac{f(\theta)}{f(\theta_0)} - \frac{g(\theta)}{g(\theta_0)} \right) \partial \pi_U(\theta)$ and, by definition of $\theta_0$,

$$\frac{f(\theta)}{f(\theta_0)} - \frac{g(\theta)}{g(\theta_0)} = \frac{g(\theta)}{f(\theta_0)} \left[ \frac{f(\theta)}{g(\theta)} - \frac{f(\theta_0)}{g(\theta_0)} \right] \leq 0.$$  \hfill \Box

Figure 6.5 shows the impact of the distribution of $L$ (specifically, the variance of $L$) on the optimal threshold and the corresponding maximum CTI. I consider for $L$ a simple hyper-exponential distribution of the second order, which allows
us to vary the coefficient of variation (COV) while keeping the mean fixed to 1. I further assume that $P_a = 0.1 e^{-a}$ and $\alpha = 0.1$. Interestingly, as the variance of $L$ increases, the optimal value of the threshold decreases while the resulting CTI increases. This can be explained as follows: by increasing the coefficient of variation of $L$, we observe few (rare) larger and larger spikes of $L$, interleaved with many smaller and smaller spikes. While large (rare) spikes of $L$ quickly fade away thanks to the exponential decay of user excitation, the presence of many small spikes allows the system to sample the user excitation at a lower level, thus yielding larger values of user response.

### 6.4.2 Real-time delivery

Recall that in this scenario the system has to make an instantaneous binary decision (selection) of impression opportunities arriving according to a Poisson process of rate $\lambda$.

The optimal selection policy can be formalized as a Markov Decision Process (MDP) over continuous space. The state of the process is given by the user excitation sampled at the time instants at which a new impression opportunity arrives. At each sampling time $t_n$, two actions $a$ are possible: either the opportunity is sent to the user ($a = 1$), or it is discarded ($a = 0$). Thus, the instantaneous reward at the generic
User Behaviour in Online Advertising

sampling time \( t_n \) is:

\[
\mathcal{R}(n, a) = \begin{cases} 
P_a(U(t_n) + L_n) & \text{if } a = 1 \\
0 & \text{if } a = 0 
\end{cases}
\] (6.11)

For tractability, we approximate the above MDP by discretizing the level of user excitation and defining a Markov Chain where the \( i \)-th state corresponds to excitation level \( U_i \). So doing, we can apply known results from the theory of stochastic dynamic programming [97, Ch. 5], which allows us to characterize the optimal filtering policy. In particular, Theorem 2.4 in [97, Ch. 5] states that, if the Markov Chain, for any possible policy, includes an ergodic state, the policy that maximizes the average reward is stationary, i.e., the action taken at a given time instant deterministically depends on the current state. Note that, in my system, the state corresponding to any arbitrarily small level \( \varepsilon \) of user excitation is ergodic since it can be reached from any other state due to the decaying behaviour of the excitation and the fact that \( L \) is assumed to have finite moments.

In general, the optimal policy can be found by solving Belman’s equation [97]:

\[
w + h(n) = \max_a \left[ \mathcal{R}(n, a) + \sum_{j=0}^{\infty} P_{ij}(a)h(j) \right], \quad n \geq 0
\]

where \( h(n) \) is a bounded function, \( w \) is a constant representing the average optimal reward and \( P_{ij}(a) \) is the probability to move from state \( i \) to \( j \) given that action \( a \) is taken. By using this approach, I found that in all cases of practical interest the optimal policy is threshold-based: impression opportunities arriving at time instants at which \( U(t_n) \leq \theta^* \), where \( \theta^* \) is an optimized threshold, have to be delivered to the user, whereas when \( U(t_n) > \theta^* \) opportunities have to be discarded.

I remark that there are cases in which the optimal policy is not threshold-based, as illustrated in the following example.

**Counterexample.** We fix \( \lambda = 2 \), \( L = 3 \), and response function \( P_a \) shown by the dotted curve in Figure 6.6. Note that the chosen \( P_a \) contains two sharp discontinuities at \( U = 6 \) and \( U = 10 \). Employing a discretization step \( \Delta u = 0.1 \), we found that the optimal policy filters out all impression opportunities arriving at time instants at which either i) \( U(t_n) > 7 \) or ii) \( U(t_n) \in [3,4.1] \), obtaining \( \text{CTI} \approx 1.03 \). The best threshold-based policy requires \( \theta^* = 7 \), and achieves \( \text{CTI} \approx 1.02 \). This is shown in
6.4 Analysis under perfect excitation information

Figure 6.6 Counterexample showing that the optimal policy for real-time delivery is not always threshold-based.

Figure 6.6, which reports the CTI achieved by the whole family of threshold-based policies, as we vary \( \theta \) (solid line). Figure 6.6 also shows the CTI achieved by the same family of threshold-based policies, modified by discarding opportunities arriving in the ‘hole’ \( U(t_n) \in [3, 4.1] \), which contains the optimal one for \( \theta^* = 7 \).

As already said, the above counterexample has to be considered an academic curiosity: in all cases that I tested where \( P_a \) is continuous, like in the exponential case, the optimal policy is threshold based. Moreover, the above counterexample suggests that possible gains achievable by non-threshold-based policies are negligible (e.g., 1.03 vs 1.02). Therefore, in the following I will focus on the performance achieved by threshold-based policies.

**Performance of threshold-based policies:** For given \( \theta \), we can compute the stationary distribution \( \pi_U(u) \) induced by the corresponding threshold-based policy. Indeed, for real-time delivery, the CTI resulting from a threshold-based policy can be related to \( \pi_U(u) \) as:

\[
\text{CTI} = \lambda \int_0^\theta P_a(u + l)f_L(l)\partial\pi_U(u).
\]
To obtain the stationary distribution $\pi_U(u)$, we derive the Chapman-Kolmogorov equations associated to the Markov process (over continuous space state) $U_i = U(t_i)$:

$$U_{i+1} = (U_i + L_i 1_{[U_i \leq \theta]}) e^{-\alpha \tau_{i+1}}$$

where $\tau_{i+1} = t_{i+1} - t_i$ and the condition in the indicator function accounts for the fact that an impression opportunity is delivered only if $U_i \leq \theta$. Now, denoted with $f_{U_i}(u)$ the pdf of $U_i$ (i.e., $f_{U_i}(u) = \frac{\partial \pi_U(u)}{\partial u}$), we can derive an integral equation relating $f_{U_i}(u)$ to $f_{U_{i+1}}(u)$. To do so, I denote with $V_i = (U_i + L_i 1_{[U_i \leq \theta]})$. By conditioning on the value $x$ assumed by $U_i$, we have:

$$f_{V_i}(u \mid x) = f_L(u - x) 1_{[x \leq \theta]} + \delta(u - x) 1_{[x > \theta]}$$

where the first and second term on the right hand side account, respectively, for the case where an impression opportunity is delivered ($x \leq \theta$) and the case where the user excitation is above the threshold. Then, unconditioning, we get:

$$f_{V_i}(u) = \int_0^\theta f_L(u - x) f_{U_i}(x) dx + f_{U_i}(u) 1_{[u > \theta]}.$$ 

Now we can observe that, conditionally over $\tau_{i+1} = t$, $U_{i+1}$ and $V_i$ are deterministically related, being $U_{i+1} = V_i e^{-\alpha t}$. Therefore, $f_{U_{i+1}}(u \mid t) = e^{\alpha t} f_{V_i}(ue^{\alpha t})$. Then,

$$f_{U_{i+1}}(u \mid t) = e^{\alpha t} \int_0^\theta f_L(ue^{\alpha t} - x) f_{U_i}(x) dx$$

$$+ e^{\alpha t} f_{U_i}(ue^{\alpha t}) 1_{[ue^{\alpha t} > \theta]}.$$ 

Finally, unconditioning we get:

$$f_{U_{i+1}}(u) = \lambda \int_0^\infty e^{(\alpha - \lambda) t} \int_0^\theta f_L(ue^{\alpha t} - x) f_{U_i}(x) dx dt$$

$$+ \lambda \int_\theta^\infty \frac{1}{\alpha} f_{U_i}(ue^{\alpha t}) e^{(\alpha - \lambda) t} dt.$$ 

(6.12)

Given the assumption on stationarity, we can impose that $f_{U_i}(u) = f_{U_{i+1}}(u) = f_U(u)$, therefore $f_U(u)$ is obtained as the only normalized solution (i.e., a solution such
that $\int f_U(u) \, du = 1$ of the following integral equation:

$$
f_U(u) = \lambda \int_0^\infty e^{(\alpha - \lambda)t} \int_0^\theta f_L(ue^{\alpha t} - x) f_U(x) \, dx \, dt$$

$$+ \lambda \int_0^\infty \frac{1}{a \log\left(\frac{a}{u}\right)} f_U(ue^{\alpha t}) e^{(\alpha - \lambda)t} \, dt. \quad (6.13)$$

6.4.3 Buffer-Driven Filtering (BDF) Strategy

I now consider a delay-tolerant delivery scenario lying in between the extreme cases of arbitrary delivery and real-time delivery. As anticipated in section 6.3, I assume that, similarly to the arbitrary delivery case, the Ad Server can send impressions to the user at arbitrary time instants (the user is supposed to be permanently exposed to ads), but within a given deadline for each ad. In other words, ads can be buffered for some time $D$ (deterministic or random) at the Ad Server. I denote by $\rho = \lambda \mathbb{E}[D]$ the corresponding ‘traffic intensity’. For such intermediate scenario, I propose a heuristic strategy, named Buffer-Driven Filtering (BDF), based on the following idea: whenever there are at least two opportunities in the buffer, we employ the optimal threshold $\theta_0$ of arbitrary delivery. When we have a single opportunity in the buffer, we employ the optimal real-time threshold $\theta^*$. Moreover, we delay the deliver of this (single) opportunity until it is close to expire, in order to maximize its acceptance probability.

I report below the pseudo-code of the BDF strategy, assuming that the optimal threshold $\theta_0$ ($\theta^*$) is available for arbitrary (real-time) delivery. The algorithm maintains two state variables: the buffer length $q$ (in number of impression opportunities), and a boolean flag saturation taking value 1 while operating as in arbitrary delivery, value 0 while operating as in real-time delivery. I assume that the system maintains an online estimate $\hat{U}(t)$ of the user excitation $U(t)$. The code shows the operations performed upon the occurrence of three main events: i) the arrival of a new impression opportunity; ii) $\hat{U}$ drops below $\theta_0$; iii) an opportunity kept in the buffer expires. Note that, when the strategy decides to send an impression to the user, it chooses the one with the earliest deadline (Earliest Deadline First, EDF), so as to maximize the buffer occupancy.
Algorithm 1  BDF strategy

Require: $\theta_0, \theta^*$

$q = 0$
$saturation = 0$
$\hat{U} \leftarrow initial \ estimate \ of \ U$

upon event $< arrival \ of \ impression \ opportunity >$ do
  if $q = 1$ then
  if $\hat{U} < \theta_0$ then
    send-impression-to-user (EDF)
    $\hat{U} \leftarrow update \ estimate \ of \ U$
    if $\hat{U} < \theta_0$ then
      saturation = 0
    else
      saturation = 1
  else
    saturation = 1
    $q \leftarrow 2$
  else
    enqueue-impression
    $q \leftarrow (q + 1)$
  upon event $< \hat{U} \ drops \ below \ \theta_0 >$ do
    if $saturation = 1 \ & \ q > 0$ then
      send-impression-to-user (EDF)
      $q \leftarrow (q - 1)$
    if $q < 2$ then
      saturation = 0
  upon event $< impression \ opportunity \ expires >$ do
    if $q = 1 \ & \ \hat{U} < \theta^*$ then
      send-impression-to-user
    else
      discard-impression-opportunity
      $q \leftarrow (q - 1)$
    if $q < 2$ then
      saturation = 0

6.4.4 Numerical evaluation

I now compare the performance of the strategies presented in the previous sections. To this end, I consider a scenario in which I fix $\alpha = 0.1$, while $L$ is assumed to be exponentially distributed (with mean 1). I consider either a monotonically decreasing
response function, or an inverse-U response function. As I vary the arrival rate \( \lambda \) of the arriving (Poisson) stream of (candidate) impression opportunities, I compare the CTI achieved by:

- the StateLess Thinning strategy (SLT), with optimal thinning probability \( p \) (section 6.3.4);
- the CTI achievable under arbitrary delivery, hereinafter denoted by \( \text{CTI}_0 = \text{CTI}(\theta_0) \), which represents an upper-bound to the system performance (section 6.4.1);
- the threshold-based real-time strategy, with optimal threshold \( \theta^* \) (section 6.4.2);
- the Buffer-Driver Filtering (BDF) strategy introduced above, for different values of traffic intensity \( \rho \).

Figure 6.7 and figure 6.8 show the CTI achieved by the considered strategies in the case of, respectively, decreasing response function \( P_a = 0.1e^{-u} \) and inverse-U response function \( P_a = 0.1ue^{-u} \). In both figures I also show on the right y axes the value of the optimal threshold \( \theta^* \) for the real-time delivery case.

Note that, consistently with what said in section 6.3.3, the upper bound \( \text{CTI}_0 \) provided by arbitrary delivery only makes sense when \( \lambda > \lambda_{\text{min}} \), where \( \lambda_{\text{min}} \) is the
rate of impression opportunities delivered to the user by the optimal policy. Such minimum rate is given by:

$$\lambda_{\text{min}} = \frac{\alpha}{\int_l \log \left( 1 + \frac{l}{\theta_0} \right) \partial F_L(t)}$$

This explains why I show \( CTI_0 \) as an horizontal line starting from the point at which \( \lambda = \lambda_{\text{min}} \). Specifically, we have \( CTI_0 = 0.0033 \) in the case of figure 6.7, for \( \theta_0 = 0.61 \), whereas we get \( CTI_0 = 0.005 \) in the case of figure 6.8, for \( \theta_0 = 1.47 \).

Results are qualitatively similar for both considered user response functions. All strategies tend to achieve similar performance for small \( \lambda \), where filtering is not needed and the best choice is to just deliver to the user all arriving impression opportunities.

As \( \lambda \) increases and filtering becomes effective, some differences arise. The CTI obtained by the threshold-based policy increases with \( \lambda \), till it saturates to the upper-bound. Note that the corresponding value of optimal threshold in the case of real-time delivery, \( \theta^* \), decreases with \( \lambda \), approaching \( \theta_0 \). The SLT strategy, instead, saturates to a lower value and much earlier than the threshold-based policy. The reason is that SLT is a simpler strategy unaware of the system state, and for large \( \lambda \)
cannot do any better than delivering a Poisson stream of opportunities to the user (which is suboptimal).

The performance of the BDF algorithm strongly depends on the traffic intensity $\rho$ (note that $\rho$ equals the average buffer occupancy of an $M/G/\infty$ queue storing the arriving opportunities). As expected, for large values of $\rho$ (e.g., larger than 10), BDF approaches the upper bound, while for small values of $\rho$ (e.g., smaller than 1) BDF essentially behaves like the real-time delivery strategy.

### 6.4.5 Trace-driven results

In section 6.3.2 I already described a simple way to fit the model to the Avazu and Outbrain traces, assuming fixed $L = 1$ and obtaining an empirical user response function with $\alpha = 0.3$. I now make an additional step, comparing the actual CTI measured on the traces with the CTI resulting from the fitted model. Specifically, for both traces I consider a simple least-square exponential fitting of the user response function (the dashed lines in figures 6.3(a) and 6.3(b)). Using the actual arrival process of impressions appearing in the traces, I build the excitation $U(t)$ of each user (assuming that users are homogeneous), setting $\alpha = 0.3$ and $L = 1$. I can then obtain the CTI as predicted by the model, and compare it with the actual CTI of the traces. Despite the strong approximations introduced by my methodology (e.g., the least-square fitting, the assumption that users are homogeneous), I found that, for both traces, the CTI predicted by the model closely matches the actual CTI, with a relative error below 5%. This further confirms the validity of the system model introduced in section 6.3.

I then tried to apply the filtering strategies to the arrival process of ads contained in the traces, in order to achieve possible better CTI. Unfortunately, the arrival rate of impressions in both traces is significantly smaller than $\lambda_{\text{min}}$, leaving little room for improvements. For example, I tried the simple SLT strategy, with a numerically optimized thinning probability $p$. The optimal $p$ turns out to be equal to 1, meaning that SLT does not lead to any improvement. Due to the scarcity of impression opportunities, the arbitrary delivery strategy cannot be applied ($\lambda < \lambda_{\text{min}}$). For the same reason, the threshold-based real-time strategy yields only marginal gains (in the order of 1%), since $U(t)$ very rarely goes above the optimal threshold $\theta^*$. 
When, instead, the BDF strategy is adopted (assuming $\rho = 100$), i.e., impressions can be delayed, and thus better spread out over time, significant improvements in CTI are obtained, namely, 7% in the case of the Avazu trace and 24% for the Outbrain trace. This can be explained by the fact that the arrival process of ads in both traces is quite bursty.

In conclusion, it can be said that: i) real traces confirm the validity of the model in representing user behaviour and predicting the resulting CTI; ii) for moderate values of impression rate, a careful optimization of the times at which impressions are submitted to the user (when this is possible, i.e., when ads can be delayed) can produce significant gains in terms of CTI.

### 6.5 Analysis under partial excitation information

Let’s now move to the case in which the Ad Server does not have perfect knowledge of the user excitation at time $t$, $U(t)$. However, the following information is available to the advertising platform: (i) the time instants at which previous impression opportunities have been delivered to the user (i.e., the sequence $\{t_i\}_{i:t_i < t}$), and (ii) the outcomes $\{X_i\}_{i:t_i < t}$, of such opportunities (i.e., whether the user has performed a valuable action ($X_i = 1$) or not ($X_i = 0$), on each opportunity). I also assume that the system is aware of the statistics of $L_i$, of $\alpha$ and of the user-response function $P_a$.

From the above information, the Ad Server can derive an estimate $\hat{U}(t)$ of the exact value of user excitation $U(t)$, to be used for deciding when/whether to send the next opportunity to the user.

Note that the best estimate $\hat{U}(t)$ is, by construction, given by $\hat{U}(t) = \mathbb{E}[U(t) \mid \{t_i\}_{i:t_i < t}, \{X_i\}_{i:t_i < t}]$. Therefore, given the structure of $U(t)$, we have:

$$\hat{U}(t) = \sum_{t_i < t} \mathbb{E}[L_i \mid \{X_j\}_{j < t}] \exp(-\alpha(t-t_i)).$$

It follows that, in order to estimate the user excitation, we have to obtain $\mathbb{E}[L_i \mid \{X_j\}_{j < t}]$. The exact analysis of $\mathbb{E}[L_i \mid \{X_j\}_{j < t}]$ is fairly difficult. For example, if we ignored the correlation between $L_i$ and $\{X_j\}_{j \neq i}$, we could compute $\mathbb{E}[L_i \mid X_i = 1]$ and $\mathbb{E}[L_i \mid X_i = 0]$. Then, by standard Bayesian analysis, in principle the distribution
of $L$ conditioned to the user’s reaction to the offered item could be obtained as:

$$F_L(l \mid X = 1) = \int_0^l \int_0^\infty P_a(u + z) \partial F_L(z) \partial \pi_U(u) \frac{\partial F_L(l)}{\partial \pi_U(u)}$$

and

$$F_L(l \mid X = 0) = \int_0^l \int_0^\infty (1 - P_a(u + z)) \partial F_L(z) \partial \pi_U(u) \frac{\partial F_L(l)}{\partial \pi_U(u)}$$

From the above expressions, it would be easy to derive $\mathbb{E}[L \mid X = 1]$ and $\mathbb{E}[L \mid X = 0]$, provided that $\pi_U(u)$ were known, which unfortunately is not.

I therefore adopt a different approach which aims at estimating directly the distribution $\pi_U^{(i)}(u)$ associated with the $i$-th impression opportunity sent to the user. To this end, we can write a recursive equation which relates $\pi_U^{(i)}(u)$ to $\pi_U^{(i-1)}(u)$, given $X_i$.

I define $U(t_i^+)$ as the user response at time $t_i^+$, i.e., right after the excitation increment that occurs upon the delivery of an impression opportunity; more formally, $U(t_i^+) = \lim_{t \downarrow t_i} U(t)$. Also, for brevity, I denote by $U_i$ and $\pi_U^{(i)}(u)$, respectively, $U(t_i^+)$ and the distribution of $U(t_i^+)$. Using such notation, we have:

$$\mathbb{P}(U_i < u \mid X_i = 1, U_{i-1} = y, \tau_i) = \mathbb{P}(U_i < u, X_i = 1 \mid U_{i-1} = y, \tau_i) \frac{\mathbb{P}(X_i = 1 \mid U_{i-1} = y, \tau_i)}{\mathbb{P}(X_i = 1)}$$

with:

$$\mathbb{P}(U_i < u, X_i = 1 \mid U_{i-1} = y, \tau_i) = \int_0^{(u-u_i^-)} P_a(u_i^- + l) \partial F_L(l)$$

where $u_i^- = y e^{-\alpha \tau}$ and $(z)^+ = \max(0, z)$. Furthermore,

$$\mathbb{P}(X_i = 1 \mid U_{i-1} = y, \tau_i) = \int_0^\infty P_a(u_i^- + l) \partial F_L(l)$$
Therefore,

\[
\pi_U^{(i)}(u \mid X_i = 1, \tau_i) = \mathbb{P}(U_i < u \mid X_i = 1, \tau_i) = \frac{\mathbb{P}(U_i < u, X_i = 1 \mid \tau_i)}{\mathbb{P}(X_i = 1 \mid \tau_i)}
\]

\[
= \frac{\int_0^\infty \mathbb{P}(U_i < u, X_i = 1 \mid U_{i-1} = y, \tau_i) \mathbb{P}(\tau_{i-1} = (y))}{\int_0^\infty \mathbb{P}(X_i = 1 \mid U_{i-1} = y, \tau_i) \mathbb{P}(\tau_{i-1} = (y))} \pi_U^{(i-1)}(y)
\]

\[
= \frac{\int_0^\infty \int_0^{u-\alpha \tau_i} P_a(ye^{-\alpha \tau_i} + l) \partial F_L(l) \partial \pi_U^{(i-1)}(y)}{\int_0^\infty \int_0^{\infty} P_a(ye^{-\alpha \tau_i} + l) \partial F_L(l) \partial \pi_U^{(i-1)}(y)}
\]

(6.14)

Similarly,

\[
\pi_U^{(i)}(u \mid X_i = 0, \tau_i) = \mathbb{P}(U_i < u \mid X_i = 0, \tau_i)
\]

\[
= \frac{\int_0^\infty \int_0^{u-\alpha \tau_i} (1 - P_a(ye^{-\alpha \tau_i} + l)) \partial F_L(l) \partial \pi_U^{(i-1)}(y)}{\int_0^\infty \int_0^{\infty} (1 - P_a(ye^{-\alpha \tau_i} + l)) \partial F_L(l) \partial \pi_U^{(i-1)}(y)}
\]

(6.15)

The above equations can be used to construct at time \(t_i^+\) an estimate of \(\pi_U^{(i)}(u)\), given \(\tau_i\), \(\pi_U^{(i-1)}(u)\) and \(X_i\). Note that, while the information about \(X_i\) is exploited to estimate \(\pi_U^{(i)}(u)\), the same information is ignored when the estimate of \(\pi_U^{(i-1)}(u)\) is built. Therefore, a natural refinement of this estimate exploits the information about \(X_i\) to obtain an a-posteriori estimate of \(\pi_U^{(i-1)}(u)\). To this end, I can write a “backward” equation that gives \(\mathbb{P}(U_{i-1} < y, X_i = 1 \mid U_i = u, X_{i-1}, \tau_i)\). First, observe that \(\mathbb{P}(U_{i-1} < y, X_i = 1 \mid U_i = u, X_{i-1}, \tau_i) = \mathbb{P}(U_{i-1} < y, U_i = u, X_{i-1}, \tau_i)\), since \(X_i\) and \(U_{i-1}\) are conditionally independent, given \(U_i\), and \(\mathbb{P}(X_i = 1 \mid U_i = u, X_{i-1}, \tau_i) = 1\). Therefore, we have:

\[
\mathbb{P}(U_{i-1} < y \mid U_i = u, X_{i-1}, \tau_i) = \frac{\mathbb{P}(U_{i-1} < y, X_{i-1} = 1 \mid U_i = u, \tau_i)}{\mathbb{P}(X_{i-1} = 1 \mid U_i = u, \tau_i)}
\]

with:

\[
\mathbb{P}(U_{i-1} < y, X_{i-1} = 1 \mid U_i = u, \tau_i) = \frac{1}{F_L(u)} \int_{(u-\alpha \tau_i)}^{u} P_a((u-l)e^{\alpha \tau_i}) \partial F_L(l)
\]
Furthermore,

\[ P(X_{i-1} = 1 \mid U_i = u, \tau_i) = \frac{1}{F_L(u)} \int_0^u P_a \left( (u-l) e^{\alpha \tau_i} \right) \partial F_L(l) \]

Then we get:

\[
\pi_U^{(i-1)}(y \mid X_{i-1} = 1, \tau_i) = P(U_{i-1} < y \mid X_{i-1} = 1, \tau_i) \\
= \int_0^\infty \int_0^y [u - ye^{-\alpha \tau_i}]^+ P_a \left( (u-l) e^{\alpha \tau_i} \right) \partial F_L(l) \partial \pi_U^{(i)}(u) \\
\int_0^\infty \int_0^y [u - l e^{\alpha \tau_i}] \partial F_L(l) \partial \pi_U^{(i)}(u)
\]

(6.16)

and similarly:

\[
\pi_U^{(i-1)}(y \mid X_{i-1} = 0, \tau_i) = P(U_{i-1} < y \mid X_{i-1} = 0, \tau_i) \\
= \int_0^\infty \int_0^y [u - ye^{-\alpha \tau_i}]^+ [1 - P_a \left( (u-l) e^{\alpha \tau_i} \right)] \partial F_L(l) \partial \pi_U^{(i)}(u) \\
\int_0^\infty \int_0^y [1 - P_a \left( (u-l) e^{\alpha \tau_i} \right)] \partial F_L(l) \partial \pi_U^{(i)}(u)
\]

(6.17)

We can therefore iterate between ((6.14) or (6.15)) and ((6.16) or (6.17)), until convergence to a fixed point is reached.

Note that, from \( \pi_U^{(i)}(u) \), we can easily obtain the distribution of \( U(t) \) for any \( t \in (t_i, t_{i+1}) \) since, by construction, we have: \( U(t) = u_i e^{-\alpha (t-t_i)} \) and, hence,

\[ P(U(t) < u) = P(U_i < u e^{\alpha (t-t_i)}) = \pi_U^{(i)}(u e^{\alpha (t-t_i)}). \]

In particular,

\[ P(U(t_{i+1}) < u) = P(U_i < u e^{\alpha (t_{i+1}-t_i)}) = \pi_U^{(i)}(u e^{\alpha (t_{i+1})}). \]

### 6.5.1 Numerical results

In the previous section I described a methodology to estimate the user excitation \( U(t) \), given the uncertainty on the sequence of \( L_i \). Aiming at showing the performance of
this method, hereinafter referred to as Feedback. I exploit (6.14)–(6.17) to compute $\mathbb{E}[U(t)]$ and use it to decide when/whether to deliver an impression opportunity to the user. I then derive the resulting CTI for both arbitrary and real-time delivery. To this end, I keep the same setting as in section 6.4.4, i.e., $\alpha = 0.1$, $L$ exponentially distributed with mean 1, and $P_a = 0.1 e^{-u}$; also, I set $\lambda = 1$ in the real-time delivery case. I compare Feedback to a simpler approach, called Fixed-$L$, according to which the evolution of $U(t)$ is estimated by assuming $L$ to be deterministic (equal to 1).

In the arbitrary delivery scenario, I apply the optimal threshold-based policy and compute the error in the estimate of the user excitation and the consequent loss in terms of CTI with respect to the case where perfect information is available (CTI$_0$). Figure 6.9(a) depicts the average value of such relative loss and highlights that Feedback gets closer to the performance achieved in the presence of perfect information.

In the case of real-time delivery, I apply the threshold-based, real-time strategy, reporting the relative gain I obtain with respect to the SLT method with optimal $p$. Figure 6.9(b) depicts such a gain, along with the results achieved when perfect knowledge of $U$ is available. Again, Feedback better approximates the case of perfect information and improves the SLT performance by almost 17%.

Then I assume that the Ad Server has inaccurate knowledge of the decay rate $\alpha$ of user excitation, namely, it underestimates $\alpha$ by 50%. Consequently, the Ad Server will compute an inaccurate value of the threshold, hence it will expose the user to impressions at non-optimal values of user excitation. The performance of Fixed-$L$ and Feedback in such scenario is illustrated in figure 6.9(a) and 6.9(b). As expected, the performance decreases with respect to the case in which the exact $\alpha$ is known; however, Feedback can better cope with the incorrect knowledge of $\alpha$ since it is able to partially compensate the errors by exploiting the information from user response.

### 6.6 Discussion and conclusions

In this chapter I have explored the novel problem of maximizing the CTI of a user subject to impressions delivered by an Ad Server, since this metric is directly related
6.6 Discussion and conclusions

Figure 6.9 Performance with different approximations of $U$ in arbitrary and real-time delivery scenarios. Note that plots (a) and (b) depict CTI relative loss and gain, respectively.
to the profit of an ad campaign, in contrast to the traditional CTR, which, by itself, is not sufficient, and can even lead to largely suboptimal profits.

Although my analysis relies on a specific user behavioural model, traces of real advertising systems have confirmed the existence of significant correlations between the click probability and the history of impressions shown to the user, which are well captured by my approach. Indeed, by fitting the parameters of the model on the considered traces, I have been able to accurately predict the CTI resulting from the temporal sequence of impressions appearing in the traces. More importantly, the model has allowed to optimize the sequence of impressions itself, achieving significant gains in terms of CTI (and thus in terms of profits).

As a first step in this new research problem, my analysis has adopted several simplifying assumptions which also have important limitations, as I briefly discuss below suggesting directions for future work.

First, I have assumed that candidate impressions are homogeneous (i.e., they are qualitatively identical). A natural extension of the analysis of this chapter would be to consider heterogeneous impressions, for example divided into different classes having their own arrival rate and user response function (one should then control the cumulative excitation received by the user).

Second, for analytical tractability I have often assumed that the arrival process of impression opportunities is Poisson. Since the activity patterns of a user are not well described by a simple Poisson process, it would be desirable to extend the analysis to more realistic processes, possibly taking into account the on-off nature (sessions) of user exposure.

Third, it would be interesting to perform a wider sensitivity analysis on the various parameters of the model, to better assess the impact of incomplete information about the user characteristics.

At last, in the real world a user is usually exposed to several competing Ad Servers and ad campaigns, and its response to them is typically not independent, but driven by cumulative effects. A game-theoretic approach would be appropriate to analyse such competition scenario.
Chapter 7

Conclusions and future directions of research

In summary, the followings are my main contributions to the research community obtained thanks to this thesis:

- I proposed a new machine learning approach for the identification of web-pages explicitly visited by users in HTTP logs collected by passive network monitors. My approach generalizes ad-hoc designed heuristics, automatically learning the patterns that characterize explicit visits, with detection precision and recall over 90% (chapter 2).

- I presented a characterization of clickstreams that differs from previous efforts for (i) covering a large population during a long consecutive period of time, and (ii) accounting for different device types used to browser the web at home (chapter 3).

- I created and contribute to the community a three-year long dataset of anonymized clickstreams, covering thousands of households in Europe. To the best of my knowledge, this is one of the largest datasets that includes clickstream graphs from regular Internet users, browsing with multiple devices (chapter 3).

- I explored techniques for users’ fingerprinting and identification, using only the domains of visited web-services. Web-services intentionally requested have proved to better characterize users (chapter 4).
• I modelled paths of users on the web, representing them in a succinct and interpretable manner. I showed that I can automatically and easily inspect the interests of users and communities, highlight transitions and diversity is such clusters of domains, and group together the interests and browsing patterns of single users and/or communities (chapter 5).

• I modelled the user interaction with recommendation systems, introducing a user behavioural model that captures the impact on performance of the history of impressions shown. The model has been validated and tuned on real traces (chapter 6).

• I improved the revenue of online advertisements systems under different scenarios by optimizing the sequence of impressions. Experimental results showed that my approach can significantly increase the revenue of an ad campaign (chapter 6).

My findings have several implications to the Internet actors. For example: (i) models of trajectories can be used to propose personalized recommendation systems for users browsing; (ii) clickstreams analysis can help advertisers to make informed decisions on whether to target ads campaigns on specific device users; (iii) network operators can find anomalies in behaviour of users or group of users; (iv) our fingerprint study should stimulate researchers to investigate privacy aspects and find countermeasures; (v) advertisement companies can use my model to optimize the sequence of impressions, achieving significant gains in terms of profits; and, more generally, (vi) my findings can help the research community to study the place of web technologies in people’s life.

I envision some possible directions for future works. In many parts of my studies I had to introduce some limiting assumptions: in order to generalize as much as possible my results, they should be removed or better studied in the evolution of these works. For example, I plan to investigate whether machine learning approaches are able to reconstruct clickstreams – at a domain level – from HTTPS traffic. Such an approach would provide a coarse view of paths followed by users while surfing the HTTPS web. Moreover, I am conceiving a set of new methods for researchers and industries in order to perform well regulated privacy preserving analytics, with data-hiding capabilities and solutions that better ensure both privacy and data availability. Refer to the conclusion part of each chapter for more details.
Thanks to the research developed in my PhD I was able to learn and cope with methodologies to perform big data analytics and modelling complex phenomena. We are now in the middle of a huge transition as the combination of low-cost computing power, ubiquitous worldwide network and advanced big data techniques is creating a digital transformation in which everything (every person and potentially every object in the world) is producing data that can be exploited to create new disruptive capabilities. Big data production by itself is not useful without its analysis, that can allow us to realize incredible transformations not only in the web, but also in our cities, energy production, manufacturing, transportation, and healthcare.

With the always increasing world population and urbanization, cities will soon become *smart-cities* in order to cope with these changes. Technology will need to improve its energy efficiency, traffic and parking management, environmental monitoring, waste management, safety and crowd control. All these areas need expertise in understanding and leverage the user behaviour obtained from the data that is already being collecting in this moment. I started applying my knowledge in data analysis, modeling and optimization in the field of smart mobility and car sharing, forecasting the future administration of electrical and autonomous car fleets.

Please refer to appendix B for more details on my published papers and my other research interests.
Appendix A

Exploitation of web navigation data: stakeholders and ethical issues

Throughout the work presented in this thesis, I used traces of web navigation data of real users. I exploited HTTP logs in chapters 2 and 3, TCP logs in chapters 4 and 5 and logs of web advertisements in chapters 6. This appendix will deepen the subject of the entities than can collect and access these kind of data, other than researchers, highlighting what are the ethical issues that arise for users, companies, scientists and governments and presenting some of the current legislation. This appendix has partly been taken from my published work in [98].

A.1 Introduction

Nowadays, almost everybody uses the web for different reasons, such as looking for news and products, accessing social networks and organizing their lives. Also in a business environment, most of the companies can not run their business without exploiting the web. For these reasons, the content that a user is consulting on the web can be classified as sensitive personal information, either if the user is a company or a private person. Thanks to the design of the web, many entities can access, partially or totally, these sensitive information. Internet Service Providers (ISPs), online services, social networks and trackers usually store as much information as possible about the users.
The data that are collected can be used in many ways: ISPs could use these information to improve their network and the quality of their services; scientists and researchers could use these data to design new services or for understanding web social implications; advertisement companies could profile users to give them specific ads; criminals could steal identities or bank information; police and public agencies could find proofs and incriminate people; companies could know what their employees are doing and decide whether to hire someone or to fire someone else.

It is clear from this partial list that the exploitation of users’ web navigation data is a very complex and delicate topic, where laws are still not comprehensive and many entities are not even aware of the current scenario. Implications are serious, from a person who exposes unconsciously his private information to an unknown third party entity, to a company that is unable to control its information to the outside world. As a result, users have lost control over their private data in the Internet.

The reminder of this appendix is structured as follow: in section A.2 I identify and deeply explain the role of all the stakeholders in the current scenario, highlighting the connections among them. In section A.3 I present the ethical issues that arise for the different groups of entities. In section A.4 I suggest some alternative scenarios, suggesting to the different entities how they could behave and some possible countermeasures to avoid improper use of users’ sensitive data. Finally, some concluding remarks are presented in section A.5.

## A.2 The stakeholders network in the current scenario

The stakeholders network has been proposed [99] as a powerful tool for analyzing and reasoning about the difficult choices within an ICT scenario. The simple construction of this network is already a good help to identify conflicts between stakeholders and missing relationships usually not considered into the specific studied landscape. The stakeholders network for the web navigation data scenario is presented in figure A.1. I will explain all the key actors of the network, as well as the connections between them.

The actors in the scenario are called entities and can be grouped in different families, based on the role they play.
A.2 The stakeholders network in the current scenario

Figure A.1 The stakeholders network involved in producing, collecting, using and controlling the web navigation data.

The first group of entity is the *users* family, depicted with green background in figure A.1. The users are the part of the network that generates data by browsing on the web; therefore they are the entities that are potentially more exposed to risks. This dangerous position is often followed by an unawareness of such risks. Indeed, as I will explain later, only a small percentage of users use some techniques to protect their data. Users family can be further divided into two entities: *private* and *corporate*. Private users generate traffic for personal reasons, e.g., web surfing, entertainment or gaming. These users do not wish to disclose their data both for privacy and for security reasons; for example, they would not share personal interests or banking information. Corporate users, instead, generate traffic for business reasons: security of these data should be fundamental for companies to avoid industrial espionage like market analysis and information about new projects.

The second group of entities is the one that could access the data, represented in yellow in figure A.1. Within this family, each entity has a different view of the data and owns them for different reasons.
The first entity is composed by the **Internet Service Providers (ISPs)**. They
give to the users the physical access to the network, therefore an ISP receive all
traffic from its users. As a consequence, ISPs are the entities that can see everything
about users behaviour, except for the encrypted traffic, i.e., HTTPS. However, even
from encrypted traffic, ISPs can still perform many user analysis, like I showed in
chapter 4 for the fingerprinting and tracking problem.

Secondly, there are the so called **first party trackers**. This entity is composed
by service providers, such as **Google** and **Facebook**. They receive some users’ data
directly from the users themselves in order to receive a particular service. For
example, an internaut that contacts **YouTube** to retrieve a video has to send the
information about the video that wish to stream in order to watch it.

The most active, and controversial entry of this group of entities is composed
by the **third party trackers**. These services are embedded into the web-sites and are
not directly linked to the original source being asked. They monetize visits and, in
practice, collect users’ personal information. Trackers use many solutions to identify
users, ranging from storing cookies on the user’s browser to tracking techniques
that fingerprint users across several web-sites [100–102]. Google’s DoubleClick,
and Yahoo YieldManager are notable examples. However, the list of companies that
build their business around information collection is of the order of several hundreds
of entries. A recent work based on passive measures [103], counted more than 400
active online tracking services. With 100 of them being regularly contacted by more
than 50% of users, and the most pervasive ones that are impossible to avoid. In
addition, in this work the authors demonstrates that 77% of users faced the first
tracker just 1 second after starting their online web activity. This demonstrates that
this phenomenon is enormously popular and it involves all Internet users.

The third group in the stakeholders network is composed by entities that wish to
use users’ data, depicted in blue in figure A.1. Inside of this family there are either
entities that could be legitimate to use some users’ data, either others that wish to
exploit them for malicious activities. As we can see from figure A.1, this family
includes all the three entities that can directly collect users’ data (ISPs, first, and
third party trackers).

Nowadays, as explained before, ISPs have the best position to collect users’ data.
However, even if they have such a privileged position, they exploit them only for few
operations. Currently, due to the birth of Software Defined Network (SDN), ISPs start
offering new services which can heavily exploit private data. Before this scenario, ISPs already exploited users’ data to extract some information to improve the users’ Quality of Experience (QoE). An example is given by the so called transparent caching or transparent proxy [104]. ISPs try to understand what are the most popular content requested in their network in order to cache such contents inside some of their servers. By applying this technique, ISPs can provide smaller latency for such contents, therefore improving users’ QoE. Clearly, this solution allows ISPs to save money as well. Indeed, storing contents inside of their servers is cheaper with respect to send the requests via other networks. In fact, ISPs have a non-marginal cost for sending data through networks not directly owned by them.

First party trackers use private information to provide some services. In addition to the needed informations, they also collect other data like cookies to allow users’ login, or to remember the basket in case of e-commerce web-sites. However, they could also exploit such information to improve their services, targeting them to a specific subset of users. For instance, many web-sites target specific e-mails for advertisement of their products or services on the basis of the users’ interest shown in their web-site.

The third party trackers are the most ambiguous since they base their business directly on the collection of personal information. Information that can be extracted either implicitly, either explicitly from the users’ web browsing activity. These companies directly use the data and could also sell them. This phenomenon is ubiquitous, with all major players and hundreds of mostly unknown companies taking part in it. The research community has focused on disclosing and quantifying the vastness of this problem [100, 101, 105, 103], but proposing just few solutions [106, 107].

As a matter of facts, profiling services are fundamental for marketing purposes, e.g., knowing products browsed on a shopping web-site, online newspapers usually read or liked movies. These information are used to deliver customized ads for the specific user. First and third party tracker themselves, other than companies that specifically buy these information, usually exploit the data for this purpose.

Another important usage of such data are police agencies and authorities: as a matter of fact in many cases they access these data for mass surveillance. Despite this practice can be positive, as it could improve the safety of people, in many cases the authorities have overstated, taking too much information for the original
purposes. An important public example of such trend is reported by Google with the Google transparency report. Indeed, by visiting this report web-page, it is possible to discover how many information have been requested by government agencies and how many of them have been actually disclosed by Google.

Also researchers in companies and universities are interested in using such data for improving the knowledge in different topics by applying data-mining and statistical techniques [108]. For example, researchers studies can be useful to find the social implications of the web or for designing better technologies to surf the web.

Last but not least, since these data are stored somewhere in data clusters, criminals could break these systems and steal huge amount of personal data. These information can illegally be used to steal identities, bank information, or passwords A historical important case of malicious activity is related to a doctor who sold a list of almost 4,000 HIV patients due to a grudge against his company.

The last group of entities in the stakeholders networks consists in the policy makers, depicted in red in figure A.1. Hard laws are mandatory policies with legal values. In this family fall UE and US government laws, as well as regional ones. Soft laws are instead the ones that are not enforced and are related to ethical issues: computer engineers, as well as general users and companies, have their own code of ethics and privacy policies.

5LinkedIn investigating reports that 6.46 million hashed passwords have leaked online, http://www.theverge.com/2012/6/6/3067523/linkedin-password-leak-online, accessed January 2018.
A.3 Ethical and social implications: open questions

As described by Walter Maner about twenty years ago [109], the use of computing technology creates, and will create, novel ethical issues that require specific studies. In this section will be presented some ethical issues and social implications that arise for the different groups of entities introduced in section A.2. I will arise some open questions, that do not have strict and definitive answers. Later on, in section A.4 I will give some advices to address these questions.

The ethical issues implicated in the users’ web data usage belong the following two domains:

1. **Privacy**: data related to navigation history are certainly confidential and sensitive. Therefore, is it possible to use and store users’ navigation data without violating their privacy? How the different entities should behave in order to respect this fundamental right?

2. **Computer crimes**: hackers can use their skills to access the stored data for malicious activities. Thus, how can entities protect these data and be sure not to put in danger users? For how long entities should store users’ data?

As we saw before, each entity has different reasons to save and use users’ data. Despite of the reasons, the first questions are related on the way they obtained the data: did they receive an explicit or an implicit user’s consent, to store and save data? Was the user aware of being tracked? Many times the answer to both question is simple: the user was not conscious of the situation, therefore he did not give any consent at all. This is often true even in presence of an explicitly form that aware the user of the data collection. For instance, many web-sites present policy terms to the users that have to be explicitly accepted. However, often users do not even read these forms, since they look standards, they are long, and not easily understandable. Moreover, most of the time the user has the perception to be constrained in order to receive the service he is interested in. As a result, how can an entity be ethically legitimate to record users’ data?

Starting from these points several other questions arise for the different entities.

- Should entities collecting users’ data share them with other entities? In the case that the second entity is a state police agency that wish to use such data
to prove a crime, is the answer the same? Even in the case where such police agency is not of the same country of the entity? In an extreme case, if in the police agency country there is the death penalty for the crime committed, what will be ethical? What should the involved entities do?

- Many companies use users’ data to profile users for advertisement goals. How deep can they go into the profiling? Is there a limit due to privacy? In this case, where is this limit and who choose it? For example, is it legitimate to send advertisements related to the health-care of a person?

- Universities and research centres desire to use as much data as they can, to be able to perform better studies. However, is it fair that they could see personal data? Who can access the database where the data are stored? For what exact purpose? For example, is it an acceptable goal to study specific users’ interests?

- A global network such as the Internet makes usual physical boundaries obsolete. How can regional hard laws face trackers that are outside their jurisdiction? How can the freedom and anarchy of the web be balanced with a centralized regional control?

These are only a limited part of the possible questions that arise by analyzing the current Internet scenario, for which fair definitive answers do not exist. However, I hope that through the increasing popularity of these questions, I can stimulate a debate between the parties that are involved in today Internet network.

### A.4 Present and future possible alternative scenarios

In section A.3 I identified the ethical issues related to users’ web navigation data. I will now show what kind of advices and suggestions I can give to the entities of the stakeholders network.

#### A.4.1 Users

An Internet user cannot avoid to use an ISP to access the Internet. However, not all the ISPs are equal. A user has the possibility, at least, to take a look at the specific
ISP policy before starting a new contract. Moreover new ISPs that want to make customer privacy their top priority are emerging.\footnote{New ISP To Make Customer Privacy Its Top Priority, http://www.themarysue.com/privacy-first-isp, accessed January 2018.}

Similarly, a user cannot simply block first-party trackers. However, if a user wants to still have a specific kind of service, he has the choice of moving to alternative service providers. For search engines, a possibility would be to use search engines that do not track and log any personally identifiable information. For example, DuckDuckGo\footnote{https://duckduckgo.com/, accessed January 2018.} does not use cookies to identify users, and it discards user agents and IP addresses from its logs. Moreover it does not even generate anonymized identifier to tie searches together. Therefore, the search engine has no way of knowing whether two searches even came from the same computer and you will get the same results as everyone else in the world. If a user still prefer Google’s search results, it can be possible to use services such as Startpage.\footnote{https://startpage.com/, accessed January 2018.} This service submits your search to Google and returns the results to you. In this case, Google sees a large amount of searches coming from Startpage servers, without knowing who originally requests each content. Whether these approaches ensure your privacy, you will never have personalized search based on your interests.

There are many possible counter-measures to avoid third party trackers to collect your data. The privacy-conscious users have easily available many privacy-enhancer browser plugins. These plugins are actually the most used protection from the tracking services. They automatically block the transmission of user’s identifying information, depending on the user’s willingness. This type of software is directly installable into web browsers and permits user to easily modify traffic, i.e. disabling cookie sending or part of javascript page, or drastically block communications to certain web services. However, these solutions is ineffective for traffic generated out of browsers (e.g., mobile application) and, additionally, the diffusion of these solutions are surprisingly very limited, as shown by Metwalley et al. [103]. Despite end-users’ concerns about privacy largely increased, motivated also by exposed government surveillance programs, Internet user does not fully grasp the extent and seriousness of the problem. To this end, a common misconception is that encryption of the web would completely protect users’ privacy. Accordingly, HTTPS usage increased by 100% each year, reaching about 50% of web flows in October 2014 [110]. In reality,
encryption increases the value of data for third party trackers. Web services that deploy encryption establish a monopoly on information by precluding any other parties from exploiting it. Moreover, HTTPS prevents third parties and malicious users to check and possibly control what kind of data is exchanged. Another possible solution for the user would be the so-called Do Not Track HTTP header.\textsuperscript{10} It is an encouraging initiative that allows users to opt out of tracking by advertising networks and analytics services. With this solution the user can choose to turn on the field in his browser, that automatically sends a special signal to the web-sites telling that the user would not like to be tracked. The main problem of this solution is that, currently, there is no consensus on how the companies you encounter should interpret the Do Not Track header. As a result, most sites do not currently change their behaviour, with few sites supporting this solution.\textsuperscript{11}

### A.4.2 Hard laws

The governments started recently to address the privacy and crime implications of the web navigation data. In 2011 the European Union \textsuperscript{111} stated that the web navigation data shall be obtained and processed fairly. This principle generally requires that a person whose data are processed to be aware of at least the following information:

- the identity of the person who is processing the data;
- the purpose or purposes for which the data are processed;
- any third party to whom the data may be disclosed;
- the existence of a right of access and a right of rectification.

Another interesting initiative of European Union is the Cookie Law.\textsuperscript{12} Thanks to this legislation, the web-sites must request consent from visitors to store or retrieve any information on a computer, smartphone or tablet. From June 2015 any website available to European visitors that uses cookies or any other technologies for non-essential tracking must:

\textsuperscript{10}\url{http://donottrack.us/}, accessed January 2018.


1. inform users that tracking technologies are used;

2. explain the reasons for using those technologies;

3. obtain the user’s consent prior to using that technology and allow them to withdraw permission at any time.

While cookies are an obvious target, the law applies to all client-side technologies used to identify an individual. Additionally, user’s consent must involve communication where the individual consciously indicates their acceptance, e.g., by clicking an icon or check box. The only exceptions are sites where tracking is strictly necessary for the provision of a service or communication requested by the user. Shopping baskets, some online applications and client-side caching to improve web-page speed would not require authorization. Instead, web-sites using analytics, advertising or customized greetings must comply.

In this context, an important problem is the difference, in terms of laws, between the European Union and the United States.\textsuperscript{13} As a matter of fact, while the EU is trying to realize a set of laws dedicated to privacy, in the US the privacy regulation is based on a self-regulatory approach, where companies provide privacy notices that make certain promises about privacy. If these promises are violated, the \textit{Federal Trade Commission} (FTC) might penalize the company. But this power generally extends only to the promises made, so a company can determine how stringently it wants to protect privacy by modulating the promises it makes. In many instances, people are given only a right to opt out of certain uses of their data, and often no right at all to limit the collection of data about themselves by certain companies. In the EU, the rules regarding individual consent for data collection, use, and disclosures are much stricter, and much more affirmative consent is required. Fortunately, United States government has started to collaborate with EU with the goal to improve users’ privacy, and the first result of this cooperation are the \textit{international safe harbor privacy principles}.\textsuperscript{14} These principles forbid to transfer personal data to non-European Union countries that do not meet the European Union directive on the protection of personal data. The aims of these principles are the protection of personal data from accidental information disclosure or loss. This task represents

\textsuperscript{13}Differences between the privacy laws in the EU and the US, \url{http://resources.infosecinstitute.com/differences-privacy-laws-in-eu-and-us/}, accessed January 2018.

the first step of a process that could lead to a common lawmaking with the aim of stopping, from the point of view of hard laws, the private information leakage.

A.4.3 Researchers and computer scientists

Computer professionals should follow an ethical code while dealing with personal information. The most famous ones are the IEEE and ACM code of ethics.\textsuperscript{15,16} In addition to these general-purpose code of ethics, recently the Oxford University propose a set of guidelines for measurement projects regarding privacy [112]. However, the problem in this scenario is not simple since researchers and engineers must face two challenges: anonymize the data used during the research and study the services that steal private information.

Regarding the first problem, some researchers [113, 114] suggested different solutions to anonymize data, while maintaining only the information that could be useful for research purpose. However, it can be very difficult to predict if in the future records in supposedly anonymized datasets will be re-identified. In general, a scientist should modify and present data that are not privacy invasive. It may be possible to find a compromise in which some level of aggregation and pre-processing to de-identify the data takes place before a dataset is collected and stored.

On the other side, research community is currently searching a way to solve privacy problems together with companies and institutions, maintaining the economic ecosystem created around usage of users’ information. Recently Telefonica, one of the most important private telecommunications companies in the world, have started an initiative called Data Transparency Lab.\textsuperscript{17} This is a collaborative effort between universities, businesses and institutions to support research in tools, data, and methodologies for shedding light on the use of personal data by online services, and to empower users to be in control of their personal data. This initiative represents a first example of how the research community can really solve privacy problems in this field.


\textsuperscript{16}ACM code of ethics, http://www.acm.org/about/code-of-ethics and www.acm.org/about/se-code, accessed January 2018

\textsuperscript{17}Data Transparency Lab, http://datatransparencylab.org/, accessed January 2018.
A.5 Conclusions

In this appendix I have discussed, from a high-level point of view, ethical issues and social implications related to the users’ web navigation data that I also used in this thesis. I showed that in the current scenario many entities collect, store, and use these information, affecting the users’ privacy. Some entities might be legitimate to access these data, e.g., researchers or police agency, while other should not have any access at all, e.g., malicious hackers. However, since the web will be even more pervasive in everyday life in the near future, the ethical implications could even enlarge their effects. I have shown some alternative scenarios and some countermeasures to mitigate the problems that arise. Firstly, users should be really aware of how their data are used and how they could improve their privacy. Secondly, policy makers are slowly trying to regulate these phenomena, but they should be capable of fast interventions. Finally, even scientists and engineers should put their force to make these data harmless for the users, carefully evaluating the implications of the data usage for their projects.
Appendix B

Papers, collaborations and prizes

In this appendix I report the list of papers published, edited or under review starting from the beginning of my PhD, i.e. November 2014. I also present my collaborations, prizes and other publications. More information can be found on https://www.tlc-networks.polito.it/public/phd-and-post-docs/luca-vassio or on my personal website https://lucavassio.wordpress.com/.

Journals:


Conferences:


Technical papers:


Previous publications (before November 2014):


In figure B.1 I report the macro-topics of my research so far.

**Research cooperations with universities and research centres:**

- Visiting student at Computer Science Department at UFMG, Belo Horizonte, from February 2017 to August 2017. Collaboration for studying dynamic of user behaviour when surfing on Internet and when using recommendation systems.

Other activities:

- Mentoring at start-up Energy Way srl for European project Climate-KIC in Modena (Italia) in June and July 2016 (32 teaching hours) about machine learning, optimization algorithms and time series analysis and forecasting.

- Visiting CAIDA research centre at UCSD (San Diego, California), July 7-9, 2016.

Travel grants obtained:


Tutored master thesis during PhD program:


Tutored thesis before PhD program:


Bibliography


