BENELEARN 2013: Proceedings of the 22nd
Belgian-Dutch Conference on Machine Learning

Preface

The first word that characterizes the organization of the 22nd edition of BENELEARN in Nijmegen is 'finally'. Although we celebrate the twenty-second edition of the Annual Belgian Dutch Conference on Machine Learning, this year marks the first time the event is held on the grounds of the Radboud University Nijmegen, on June 3, 2013. The 90-year old university, located in the oldest city of the Netherlands, hosts a thriving community of machine-learning researchers, spread over several departments and research centers on its campus. This is a typical trend for machine learning: although a strong component of the AI and computer science curriculum, its application has become so widespread that you can be a full-time machine learning researcher at the Faculty of Arts or at a neuroscience research institute.

BENELEARN continues to serve as one of the main occasions for regional machine learning scientists to present their work and foster new collaborations. We opted for a one-day event, filled with oral and poster presentations and a keynote lecture. We are very happy to welcome prof. Dan Roth as our invited speaker. Dan Roth is a Professor in the Department of Computer Science and the Beckman Institute at the University of Illinois at Urbana-Champaign. He is a Fellow of the ACM, AAAI and ACL, for his contributions to Machine Learning and to Natural Language Processing. He has published broadly in machine learning, natural language processing, knowledge representation and reasoning, and learning theory, and has developed advanced machine learning based tools for natural language applications that are being used widely by the research community. Prof. Roth will talk about constrained conditional models and their application to text understanding.

We received 37 submissions divided in two categories: full, original research papers and compressed contributions summarizing work that has been peer reviewed and accepted for publication elsewhere. We accepted 33 papers for presentation. Of these, 12 are original research papers, and 21 are compressed contributions. The programme features 12 oral presentations and 21 poster presentations.

We would like to extend our thanks to the programme committee for performing an outstanding and timely job in the reviewing process, and in particular to Maarten van Someren for his steering and mentoring, and to Willem Waegeman for kindly sharing information and experiences from last year's BENELEARN. We are indebted to our local organization team consisting of Nicole Messink, Tom Claassen, Florian Kunneman, Joris Mooij, Rahim Saeidi, Suzan Verberne, and Sicco Verwer. We would also like to mention our sponsors who made this conference possible: NWO, SIKS, and Textkernel.

Finally, we thank you for attending BENELEARN and enriching it with your contribution. We wish you a fruitful day.

Antal van den Bosch, Tom Heskes, David van Leeuwen
BENELEARN 2013 Program Chairs

Table of Contents

BENELEARN 2013: Front Matter

BENELEARN 2013: Invited speaker

BENELEARN: Papers

BENELEARN 2013: Abstracts

BENELEARN 2013 Program Committee

Thomas Abeel	Ghent University
Michael Biehl	University of Groningen
Hendrik Blockeel	K.U. Leuven
Sander Bohte	CWI
Gianluca Bontempi	Universite Libre de Bruxelles
Toon Calders	Eindhoven University of Technology
Walter Daelemans	University of Antwerp
Jesse Davis	KU Leuven
Luc De Raedt	Katholieke Universiteit Leuven
Kris Demuynck	K.U. Leuven
Tom Dhaene	Ghent University
Damien Ernst	University of Liege
Ad Feelders	Universiteit Utrecht
Jort Gemmeke	K.U. Leuven
Pierre Geurts	University of Lige
Bart Goethals	University of Antwerp
Peter Grünwald	CWI
Michele Gubian	Centre for Language and Speech Technology, Radboud University Nijmegen
Tom Heskes	Radboud University Nijmegen
Katja Hofmann	ISLA, University of Amsterdam
Bert Kappen	Radboud University
Jan Lemeire	Vrije Universiteit Brussel
Marco Loog	Delft University of Technology
Bernard Manderick	COMO Lab. Vrije Universiteit Brussel
Joris Mooij	Radboud University Nijmegen
Ann Nowe	VUB
Mykola Pechenizkiy	Department of Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands
Mannes Poel	University of Twente/Dept. Computer Science
Eric Postma	TiCC, Tilburg University
Stephan Raaijmakers	VU University
Jan Ramon	K.U.Leuven
Rahim Saeidi	Radboud University Nijmegen
Yvan Saeys	Department of Plant Systems Biology, Flanders Institute for Biotechnology (VIB), c
Lambert Schomaker	University of Groningen
Khalil Simaan	University of Amsterdam
Evgueni Smirnov	UM
Johan Suykens	K.U. Leuven, ESAT-SCD
Evgeni Tsivtsivadze	The Netherlands Organization for Applied Scientific Research (TNO)
Karl Tuyls	Maastricht University

Stefan Van Aelst	Ghent University
Dirk Van Compernolle	Katholieke Universiteit Leuven
Antal van Den Bosch	Radboud University Nijmegen
Laurens Van Der Maaten	Tilburg University
Peter Van Der Putten	LIACS, Leiden University
Pegasystems	
Marcel van Gerven	Donders Institute, Radboud University Nijmegen
David van Leeuwen	Radboud University Nijmegen
Martijn Van Otterlo	Radboud University Nijmegen
Maarten Van Someren	University of Amsterdam
Menno Van Zaanen	Tilburg University
Joaquin Vanschoren	K.U.Leuven
Cor Veenman	Netherlands Forensic Institute
Celine Vens	K.U.Leuven
Suzan Verberne	Radboud University Nijmegen
Michel Verleysen	Universite catholique de Louvain
Jilles Vreeken	Universiteit Antwerpen
Willem Waegeman	Ghent University
Wim Wiegerinck	Radboud University
Marco Wiering	University of Groningen
Pascal Wiggers	T.U. Delft

BENELEARN 2013 Organizing Committee

Tom Claassen
Florian Kunneman
Nicole Messink
Joris Mooij
Rahim Saeidi
Suzan Verberne
Sicco Verwer

BENELEARN 2013: Author Index

BENELEARN 2013: Invited Speaker

Dan Roth, Computer Science and the Beckman Institute University of Illinois at Urbana/Champaign: Constrained Conditional Models: Towards Better Semantic Analysis of Text

Computational approaches to problems in Natural Language Understanding and Information Extraction are often modeled as structured predictions predictions that involve assigning values to sets of interdependent variables. Examples include semantic role labeling (analyzing natural language text at the level of who did what to whom, when and where), syntactic parsing, Identifying events, entities and relations in natural language text, transliteration of names, and textual entailment (determining whether one utterance is a likely consequence of another). Over the last few years, one of the most successful approaches to studying these problems involves Constrained Conditional Models (CCMs), an Integer Learning Programming formulation that augments probabilistic models with declarative constraints as a way to support such decisions. I will present research within this framework, discussing old and new results pertaining to inference issues, learning algorithms for training these global models, and the interaction between learning and inference.

Short Bio

Dan Roth is a Professor in the Department of Computer Science and the Beckman Institute at the University of Illinois at Urbana-Champaign and a University of Illinois Scholar. He is the director of a DHS Center for Multimodal Information Access Synthesis (MIAS) and holds faculty positions in Statistics, Linguistics, and at the School of Library and Information Sciences.

Roth is a Fellow of the ACM, AAAI and ACL, for his contributions to Machine Learning and to Natural Language Processing. He has published broadly in machine learning, natural language processing, knowledge representation and reasoning, and learning theory, and has developed advanced machine learning based tools for natural language applications that are being used widely by the research community.

Roth is the Associate Editor-in-Chief of the Journal of Artificial Intelligence Research (JAIR) and will serve as Editor-in-Chief for a two-year term beginning in 2015. He was the program chair of AAAI11, ACL03 and CoNLL'02, has been on the editorial board of several journals in his research areas and has won several teaching and paper awards. He has also given keynote talks and presented tutorials in some of the major conferences in his research areas.

Prof. Roth received his B.A Summa cum laude in Mathematics from the Technion, Israel, and his Ph.D in Computer Science from Harvard University in 1995.

BENELEARN 2013: Papers

A query language for constraint-based clustering

Antoine ADAM ANTOINE.ADAM@CS.KULEUVEN.BE
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, 3001 Leuven, Belgium

Hendrik Blockeel HENDRIK.BLOCKEEL@CS.KULEUVEN.BE
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, 3001 Leuven, Belgium
Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands

Keywords: query language, inductive database, constraint-based clustering

Abstract

Clustering is a widely used data mining task and a lot of constraint-based clustering methods have been developped. Our work focus on the problem of integrating constraint-based clustering in an inductive database system. We propose a new extension of SQL for constraint-based clustering. We present a concrete application in the context of microbiology.

1. Introduction

Data mining provides many methods to discover patterns or learn models from data. Even for a given type of pattern, say, association rules, multiple systems are usually available, each with their strengths and weaknesses in terms of efficiency and flexibility. As a result, it may not be easy, even for specialist users, to solve a particular data mining problem in the best possible way. Inductive database (IDB) systems (Imieliński & Mannila, 1996) can address this problem to some extent.

The idea behind IDBs is to seamlessly incorporate data mining in databases. In an IDB system, patterns are first-class citizens and can be manipulated at the same level than the data using an inductive query language. An interesting characteristic of such a query language is that it is declarative. This means the user specifies the task to execute (e.g., "find all association rules with confidence at least c, support at least s, and one of A, B, C in the head"), but not the algorithm or method that should be run. This is an advantage over data mining tools such as Weka (Hall et al., 2009), where the user must select a particular algorithm, and then accept the limitations that this choice entails. In a declarative specification, it is also easier to specify constraints on the patterns to look for. This links inductive databases to constraint-based data mining, as argued by Džeroski (2011).

Much research on inductive databases has focused on one specific data-mining task, namely association rule mining. Certain other tasks, among which clustering, have not received much attention. Further, for those IDB systems that do cover different types of tasks, the formulation of a problem can be difficult for non-specialists.

In this work, we focus on the problem of constraint-based clustering or semi-supervised clustering. Various methods have been developed in this field, introducing various kinds of constraints that act at different levels. It can go from global-level constraints that act on the resulting clusters, for example specifying the number of clusters or a minimum size of the cluster, to instance-level constraints, for example must-link and cannot-link constraints studied by Wagstaff (2002), also called equivalence constraints, that state if two instances must be or cannot be in the same cluster. We propose a query language, an extension of SQL, that allows the user to query for clusterings and formulate constraints in an easy way.

The remainder of this paper is organized as follows: Section 2 discusses related work; Section 3 presents the query language we propose; Section 4 shows an application of the query language in a microbiology context; Section 5 concludes and discusses future work.

Appearing in *Proceedings of BENELEARN 2013*.

2. Related work

The concept of inductive database has been introduced by Imielinski and Manilla (1996), followed then by De Raedt (2002b). Boulicaut and Masson (2005) and more recently Romei and Turini (2011) discussed some requirements of inductive database languages and compare some existing systems.

Various inductive database systems have been developped for different types of databases: SQL-based systems (MINE RULE (Meo et al., 1998), DMQL (Han et al., 1996), MSQL (Imieliński & Virmani, 1999), SPQL/ConQueSt (Bonchi et al., 2006), Mining Views (Blockeel et al., 2012), SiQL/SINDBAD (Wicker et al., 2008), ATLaS (Wang & Zaniolo, 2003)), DMX (Microsoft, 2012); XML-based systems (XMINE RULE (Braga et al., 2003), KDDML (Romei et al., 2006)); and logic-based systems (RDM (De Raedt, 2002a), LDL++ (Giannotti et al., 2004)). Many of these systems only deal with association rule mining (MINE RULE, DMQL, MSQL, XMine). The SINDBAD system, along with the SiQL language, supports the whole data mining process, from pre-processing to post-processing. It includes clustering but only using the k-Medoids algorithm. The ATLaS system extends SQL with user-defined aggregates which can be used for clustering. DMX also includes clustering with KMeans and EM algorithms. The Mining Views framework does not extend the query language but uses virtual mining views to do different data mining tasks. It has the advantage to integrate the data mining process in the database management system. There are also applied system specific to one domain, like Molfea (Helma et al., 2002), for mining frequent molecular structures. Even if some of these language can be extended to include equivalence constraints for clustering, we propose a new language to express such constraints in an easy and declarative way that fits the SQL relational model.

3. Query language

In this section, we present our query language. Before presenting the actual query, we will discuss the principles that guided our choice in the design of the query language.

3.1. Language properties

The goal of the project is to build a system that allows someone who does not necessarily know data mining to apply data mining algorithms on his data. For this the user should be able to ask what he wants in an easy and declarative way. We list different characteristics that we want our language to have and that guided the design of the query language.

- First, the language should be *concise*. This means queries for simple problem should be short. For instance, asking for a simple clustering without any constraints or specific parameters should be very easy.
- Second, the language should be *intuitive*, and this for both formulating and understanding a query. This means that given a question, it should be easy to formulate it in a query, but also that a written query can be easily understandable.
- Finally, the language should be *expressive*. This means that the language should allow formulating complex queries. This implies that the language should allow various constraints that can be combined. In the idea of an iterative knowledge discovery process, this also includes the closure principle. This principle says that we should be able to query the result of a query. This allows composing queries in a complex way. For instance, it is possible to first execute a clustering using some parameters, then select the instances of one cluster and do another clustering on these instances using other parameters.

3.2. Language syntax

We choose to build an extension of SQL as it is widely used and intuitive. In the logic of inductive database, data and patterns should be considered at the same level. This implies that a query for patterns should follow the same logic as a query for data. For this reason, we designed our clustering query based on the SELECT query of SQL.

For a better understanding, we will show query examples on the following imaginary dataset : we have a table named *points* with 4 attributes *id*, x, y, *valid*. *id* is an integer, x and y are real numbers, *valid* is 0 or 1 or null if unknown.

3.2.1. CLUSTER statement

We introduce the new CLUSTER statement in Figure 1. Let us explain the different parts :

data The data we want to cluster. It is a table whose columns are the attributes and lines are the instances to cluster. It can be a table present in the database or the result of a SQL select query.

attributes This part differs from the SELECT query. In the select query, it indicates which column to

```
<statement> ::= "CLUSTER" <attributes>
  "FROM" <data>
  ["WITH" <constraints>]

<attributes> ::= "*" | <attr>
<attr> ::= <attributename>[, attr]

<constraints> ::= <c> ["AND" <constraints>]
<c> ::= <nbclusterconstraint>
      | <linkconstraint>

<nbclusterconstraint> ::=
  "NumberOfClusters = n"

<linkconstraint> ::=
  ["SOFT"]["MUST"|"CANNOT"] LINK
  <cdata>
  ["BY "<attributename>]
```

Figure 1. CLUSTER statement

select and return in the result. In a cluster query, this indicates which columns to use for the clustering task. This way, you can ignore some attributes that are not relevant for the clustering task but still have them in the result of the query, like the Ids. As in the SELECT query, a * means that all columns are used for the clustering.

constraints : the constraints you can add to the clustering. They will constrain the result but also the method used to solve the query. The constraints available are :

- the number of clusters : to specify the number of cluster wanted.
- link constraints: instance-level constraints to specify if some instances must be or cannot be in the same cluster.

Here is an example of a query to cluster all the valid points according to x and y.

```
CLUSTER x,y
    FROM (SELECT * FROM points
          WHERE valid=1)
```

3.2.2. Link constraints

Let us now look at the syntax of the link constraints.

cdata The constrained data following LINK are the instances involved in the link constraint. The set of the constrained instances must be a subset of the data selected in the CLUSTER query. Subsequently, they could in principle be selected by a query: SELECT * FROM (data) WHERE (conditions). However, to be more concise and to avoid repeating the (data) part, we only put the conditions that would follow the WHERE in such a query.

MUST/CANNOT LINK The idea of this constraint was first to incorporate must-link and cannot-link constraints as formulatied by Wagstaff (2002). These constraints are pair-wise constraints which means one concerns only two instances. It can be easily understood that if one wants to specify a lot of constraints, the size of one query can quickly increase too much. To specify quickly a large number of constraints, we allow the data to be composed of more than two instances. For MUST LINK, it is supposed that all instances in *cdata* must be in the same cluster. For CANNOT LINK, it is supposed that all instances in *cdata* have to be each in a different cluster. This allows specifying small group of instances that should be clustered together in a more concise way than specifying all pair-wise constraints. Bar-Hillel et al. (2006) studied this idea of making small groups of instances, that they call chunklets. In the following example, the problem is to cluster all the points in 2 cluster, knowing that the points 3, 4 and 6 must be in the same cluster and the points 1 and 3 must be in different clusters. The query formulating this problem is as follows:

```
CLUSTER x, y
  FROM (SELECT * FROM points)
  WITH MUST LINK (id IN (3, 4, 6))
  AND CANNOT LINK (id IN (3, 1))
```

BY The BY word can be used with MUST LINK and CANNOT LINK to add link constraints between instances using an attribute of the data. With MUST LINK, must-link constraints will be created between each pair of instances that have the same value for the specified attribute. With CANNOT LINK, cannot-link constraints will be created between each pair of instances that have different values for the specified attribute. To avoid repetition, it is also possible to just say LINK (data) BY attribute. This will create must-link and cannot-link constraints according to the specified attribute as if it was MUST LINK (data) BY attribute AND CANNOT LINK (data) BY attribute. This allows for more concise queries

for instance in the case of partially labeled data. However, if no BY statement is present after the LINK, it is considered as MUST LINK. In the next example, some of the points are labelled as valid, some as invalid, and the rest is unknown. The problem is to cluster all the points in 2 clusters with the valid ones in one cluster and the unvalid ones in another cluster.

```
CLUSTER x, y
  FROM (SELECT * FROM points)
  WITH LINK (valid=0 OR valid=1)
    BY valid
```

SOFT The link constraints are considered hard constraints by default. However, one may be also interested in soft constraints. For instance, let us suppose there is some label that makes a partition of the data. Using this partition as a bias, one can be interested in clustering the data using other attributes do that instances having the same label are more likely to be in the same cluster. This can be achieved by adding a SOFT LINK constraint using this label. In the following example, all points are known as valid or invalid. The problem is to cluster the points according to x and y with a preference that the valid instances are in the same cluster and the unvalid ones are in another one.

```
CLUSTER x, y
  FROM (SELECT * FROM points)
  WITH SOFT LINK BY valid
```

3.3. Result of a query

One of the principles of database and querying language is the closure principle. It says that the result of a query should be queryable. This allows an iterative process for exploring the data progressively. As we are querying tables, the result should also be a table. As a first solution, we adopt one solution of SINDBAD (Wicker et al., 2008). The result of the query is the input table (data) where a column *cluster* has been added that contains for each instance its cluster assignement as a strictly positive integer. Figure 2 shows the different elements of the query CLUSTER x, y FROM (SELECT * FROM points).

This is a very simple way of giving the result of a clustering algorithm. However, it only works for partitioning clustering. It does not allow presenting the result of hierarchical, density-based, overlapping or other kinds of clustering. Such methods can still be used to build the clusters but the result should be a partition of the data. It is an issue we want to addres in future work.

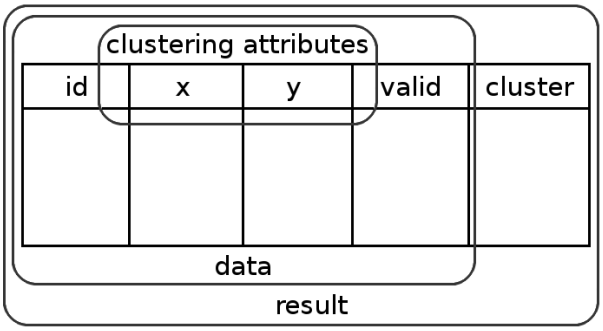


Figure 2. The cluster query viewed as table.

3.4. Query execution

Once a query has been formulated, it has to be executed. The problem is then to choose what method to use. Different methods or algorithms may not lead to the same result. Besides, we have to take into account the different constraints we can add. To do so, we decided to execute different methods depending on the constraints specified. The choice of the method is done as follows:

- By default, the distance used is a Euclidian distance. If soft link constraints are specified, a metric is learned and the resulting Mahalanobis distance is used as in (Bar-Hillel et al., 2006).
- If there are hard link-constraints, the CopKmeans algorithm (Wagstaff, 2002) is executed. For now, the number of clusters has to be specified. If it is not, it is set to the minimum number of clusters so that the link constraints can be respected. If this number is 1, it is set to 2.
- If there are no link constraints, the EM algorithm is used and if the number of clusters is not specified, cross-validation is used to determine it. We used for this the weka implementation of EM.

Currently, only these algorithms are implemented as we have focused our work on the formulation of the query but not on its execution. One of the next goals in our future work is to improve this execution. The challenge will be to combine different kind of constraints.

4. Application

In this section, we present how the query language is used in a concrete application. Our language is domain independent. We here show an example of how it can be integrated in software specific to a domain, in this case microbiology.

4.1. Cellphinder project

Our work is part of a project called CellPhinder. This project groups researchers from microbioloby and computer science and it is the initial source of motivation for our new query language. Our role in the project is to make data-mining available for the microbiologists. As said earlier, our query language is declarative. Therefore, it can be used by people who do not know clustering algorithms but are interested in the result. We indeed believe it will be easier to learn how to use our language than to learn various clustering methods and to have to put the data into the right format. We also choose to use a query language because it easily allows an iterative exploration of the data by reusing previous queries to build new ones. As the microbiologists are the first user of our system, we choose the patterns and constraints that are useful and of interest to them. This is why we focused on clustering, as it provides two interesting learning patterns: on one hand, the clusters can identify normal behavior in the data; on the other hand, instances far from any clusters or in an isolated cluster are outliers, which is also of interest for the microbiologists.

4.2. The software

We are currently building software to make our language accessible to microbiologists. It will provide an easy access to the data, as well as visualisation tools, for example distribution graphs. The user will have the possibility to build the query step by step using graphic tools but also to directly type the query he wants to execute. The software will show the result of the query in various representations, textual or graphical. At this moment, the software is still being developed. When the microbiologists are able to use the software, they will provide us with direct feedback on the language.

Our software is implemented in Java. The system is coupled with a MySQL database. When a cluster query is parsed, the SQL part is given to the database and the resulting data is given to the clustering engine. This engine uses Weka implementations of KMeans and EM and our custom-made implementation of Cop-KMeans.

4.3. Query examples

4.3.1. The data structure

The structure of the data we are working on is as follows. An Experiment consists of various lineages. A Lineage is the set of cells that originates from one mother cell. This mother cell grows and then divides in two cells, which grow and each divide in two cells, etc... From a Cell, different parameters are measured at each definite lapse of time: length, width, curvature, perimeter, growthspeed,... A cell then has different states over time. Finally, Fluorescence spots are also measured for each state. Fluorescence is made by proteins inside the cell. The fluorescence can be diffuse or localised. Consequently, we have a database made of five tables: experiment, lineage, cell, stateovertime, fluorescence. These tables have one-to-many relation from one to the next (an experiment has many lineage, that have many cells...). We also have a relation inside the cell table between parent/children cells.

4.3.2. Example 1

Assume data from a new experiment, experiment number 5, is available. The different id parameters in the following examples are not relevant for the example, they are only here to have an exact, realistic query. First, we want to cluster the cells of the experiment.

```
CLUSTER LifeTime, LagTime, LengthMean
  FROM (SELECT c.Id, c.LifeTime, c.LagTime,
               AVG(s.Length) AS LengthMean
        FROM stateovertime s, cell c,
             lineage l
        WHERE l.ExperimentId=5
          AND c.LineageId = l.Id
          AND s.CellId = c.Id
          GROUP BY c.id)
```

4.3.3. Example 2

By looking at the data, a few lineages have been found that seem to behave similarly. Another one has been found that seems to behave really differently than the others. It can be interesting to know if there are other lineages like this in the data. Must-link constraints can be added between the "normal" lineages and cannot-link constraints between the special and the others.

```
CLUSTER LifeTimeMean, LagTimeMean
  FROM (SELECT l.Id,
          AVG(c.LifeTime) AS LifeTimeMean,
          AVG(c.LagTime) AS LagTimeMean,
        FROM cell c, lineage l
        WHERE c.LineageId = l.Id
```

```
        GROUP BY l.id)
    WITH MUST LINK (Id IN (20, 21,
                           22, 23, 25))
      AND CANNOT LINK (Id IN (20, 24))
```

4.3.4. Example 3

The whole dataset is divided between mutants, which are cells that have been modified, and wildtypes, which are unmodified cells. This suggests certain similarity between cells that are of the same type and it may be interesting to use this background knowledge to do clustering. However, it is already a full partition of the data thus hard constraints are not useful. Therefore, soft constraints can be used. In the lineage table, there is a *Mutant* attribute that is 0 is the cells of the lineage are wildtypes and 1 if they are mutants.

```
CLUSTER LifeTime, LagTime,
    LengthMean, WidthMean
  FROM (SELECT c.Id, c.LifeTime,
          c.LagTime, l.Mutant
          AVG(s.Length) AS LengthMean
          AVG(s.Width) AS WidthMean
        FROM stateovertime s, cell c,
            lineage l
        WHERE l.ExperimentId=5
          AND c.LineageId = l.Id
          AND s.CellId = c.Id
        GROUP BY c.id)
  WITH SOFT LINK BY Mutant
```

5. Conclusions & Future work

In our work, we have considered must-link and cannot-link constraints which are instance level constraints. The next step will be to include more constraints in our language: feature-level constraints (so that some features can be specified more important than others), global-level constraints (minimum cluster size, balanced clusters). This will raise the problem of solving the query. Indeed, there exist algorithms to solve clustering with these different type of constraints separately. However, the problem occurs when combining different types of constraints in one query.

Another issue we want to adress is the problem of the representation of the result. Indeed, we can only present partitioning clusterings but it can be interesting to try to include other types of results like overlapping, hierarchical or model-based clusterings.

Finally, we have currently included outlier detection as a post-processing step of clustering in our software but it can be interesting to include it in the language as a real task.

Acknowledgments

For their useful comments on this paper, we would like to thank the reviewers as well as Tias Guns. For their collaboration in the Cellphinder project, we would like to thank Abram Aertsen and Sander Govers from the Laboratory of Food Microbiology of the KULeuven. This work is funded by the KULeuven in the project IDO/10/012: Elaboration of the CellPhInDER platform.

References

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2006). Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, *6*, 937.

Blockeel, H., Calders, T., Fromont, É., Goethals, B., Prado, A., & Robardet, C. (2012). An inductive database system based on virtual mining views. *Data Mining and Knowledge Discovery*, *24*, 247–287.

Bonchi, F., Giannotti, F., Lucchese, C., Orlando, S., Perego, R., & Trasarti, R. (2006). Conquest: a constraint-based querying system for exploratory pattern discovery. *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on* (pp. 159–159).

Boulicaut, J.-F., & Masson, C. (2005). Data mining query languages. *Data Mining and Knowledge Discovery Handbook*, 715–726.

Braga, D., Campi, A., Ceri, S., Klemettinen, M., & Lanzi, P. (2003). Discovering interesting information in xml data with association rules. *Proceedings of the 2003 ACM symposium on Applied computing* (pp. 450–454).

De Raedt, L. (2002a). Data mining as constraint logic programming. *Computational Logic: Logic Programming and Beyond*, 113–125.

De Raedt, L. (2002b). A perspective on inductive databases. *ACM SIGKDD Explorations Newsletter*, *4*, 69–77.

Džeroski, S. (2011). Inductive databases and constraint-based data mining. *Formal Concept Analysis*, 1–17.

Giannotti, F., Manco, G., & Turini, F. (2004). Specifying mining algorithms with iterative user-defined aggregates. *Knowledge and Data Engineering, IEEE Transactions on*, *16*, 1232–1246.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, *11*, 10–18.

Han, J., Fu, Y., Wang, W., Koperski, K., Zaiane, O., et al. (1996). DMQL: A data mining query language for relational databases. *Proc. 1996 SiGMOD* (pp. 27–34).

Helma, C., Kramer, S., & De Raedt, L. (2002). The molecular feature miner MolFea. *Proceedings of the Beilstein-Institut Workshop*.

Imieliński, T., & Mannila, H. (1996). A database perspective on knowledge discovery. *Communications of the ACM*, *39*, 58–64.

Imieliński, T., & Virmani, A. (1999). MSQL: A query language for database mining. *Data Mining and Knowledge Discovery*, *3*, 373–408.

Meo, R., Psaila, G., & Ceri, S. (1998). An extension to sql for mining association rules. *Data Mining and Knowledge Discovery*, *2*, 195–224.

Microsoft (2012). Data mining extensions (DMX). `http://msdn.microsoft.com/en-us/library/ms132058.aspx`. Accessed on 30-April-2013.

Romei, A., Ruggieri, S., & Turini, F. (2006). KDDML: a middleware language and system for knowledge discovery in databases. *Data & Knowledge Engineering*, *57*, 179–220.

Romei, A., & Turini, F. (2011). Inductive database languages: requirements and examples. *Knowledge and Information Systems*, *26*, 351–384.

Wagstaff, K. L. (2002). *Intelligent clustering with instance-level constraints*. Doctoral dissertation, Cornell University.

Wang, H., & Zaniolo, C. (2003). Atlas: A native extension of sql for data mining. *Proceedings of the 3rd SIAM International Conference on Data Mining*.

Wicker, J., Richter, L., Kessler, K., & Kramer, S. (2008). SINDBAD and SiQL: An inductive database and query language in the relational model. *Machine Learning and Knowledge Discovery in Databases*, 690–694.

An analytical approach to similarity measure selection for self-training

Vincent Van Asch VINCENT.VANASCH@UA.AC.BE
CLiPS Research Centre, University of Antwerp, Prinsstraat 13, 2000 Antwerpen, Belgium

Walter Daelemans WALTER.DAELEMANS@UA.AC.BE
CLiPS Research Centre, University of Antwerp, Prinsstraat 13, 2000 Antwerpen, Belgium

Keywords: similarity measures, domain knowledge, machine learning, part-of-speech tagging, self-training

Abstract

We present a framework for investigating properties of similarity measures as a criterion for selecting the best-suited measure for a specific task, in this paper: corpus selection for self-training. We focus on the squared Pearson's correlation coefficient as the property to rank similarity measures. Self-training is an unsupervised domain adaptation technique, in which three corpora are involved. Especially, the choice of the unlabeled corpus can be important and we show that similarity measures can be helpful when selecting an unlabeled corpus. In addition, we found that the correlation coefficient between similarity and accuracy of a similarity measure can be used to select the most suitable similarity measure, but other properties of similarity measures do also play a role.

1. Introduction

We first give a definition of *similarity measure*, since it is a vague term. In the context of this paper, a similarity measure is any function that produces a real number when applied to two text corpora. The output of the function should never switch sign and when two corpora are more similar, the absolute value of the similarity measure should be smaller. Divergences, like the Kullback-Leibler divergence, can be used as similarity measure, but divergences are certainly not the only candidates. When the two corpora are from different domains, the similarity measure can be called a domain similarity measure.

Domain similarity measures have been used in different natural language processing (NLP) setups (Zhang & Wang, 2009; McClosky, 2010; Plank, 2011) and, in general, the best-suited similarity measure depends on the task and the specific function of the similarity measure. Also combinations of different similarity measures have been tried. Nevertheless, it remains unclear which properties of a good similarity measure are responsible for its superiority. Our hypothesis is that a limited set of relevant properties exists and, depending on the processing task, some of the properties become more important than others. If this point of view is correct, creating an overview of existing similarity measures and their ranking for the different properties, would liberate the researcher from having to try all similarity measures and all combinations of measures in order to find the most appropriate measure(s).

In this paper, we investigate one candidate property, namely the degree of linear correlation between the similarity between two corpora and the accuracy in a machine learning experiment, using one of the corpora as the training corpus and the other corpus as the test corpus. The incentive to focus on the linear correlation comes from a general observation in domain adaptation literature: the more the domains of the test and the training corpus resemble each other, the better the performance of a machine learner will be. In addition, it has been found that for part-of-speech tagging, the correlation between accuracy and similarity is indeed linear (Van Asch & Daelemans, 2010).

When the linear correlation is selected as the discriminative property for similarity measures, it is possible to define what *best-suited* signifies. In this isolated situation, the *best-suited* similarity measure is the measure

Appearing in *Proceedings of BENELEARN 2013*.

that exhibits the highest squared Pearson correlation coefficient, r^2 (Pearson, 1896). With circular reasoning, this means that performance is the best similarity measure, because in this case the linear correlation would be perfect. Indeed, the best way to find out the result of an experiment is by running that experiment, but running an experiment can be time-consuming or there may be no annotated data available to actually compute the performance. For this reason, similarity measures that can be more quickly computed and that do not require annotated data are investigated. The linear correlation of these measures will be less strong, but, in return, they come with annotation-independence.

The remainder of this paper consists of an overview of the related research (Section 2), definitions of the different similarity measures that are used (Section 3), a presentation of the machine learning task (Section 4), the concept of self-training and the performance indicator (Section 5), experimental results (Section 6). A final section contains the conclusions and perspectives.

2. Related research

Divergences are used in natural language processing in various situations ranging from feature selection and training corpus creation to measuring the similarity between two language models (Della Pietra et al., 1997; Lee, 2001; Gao et al., 2002; Daumé III & Marcu, 2006; Chen et al., 2009; Mansour et al., 2009; Zhang & Wang, 2009; McClosky, 2010; Moore & Lewis, 2010; Plank, 2011). Some of the divergences that are used are perplexity, Kullback-Leibler divergence, and the Rényi divergence.

It is possible to use the divergence as such, using its value to draw inferences about corpora (Verspoor et al., 2009; Biber & Gray, 2010), but the most interesting usages apply the divergence to a machine learning system. A good example of such an application is the prediction of parsing accuracy (Ravi et al., 2008).

Despite the fact that authors have shown that a divergence (Van Asch & Daelemans, 2010; Plank, 2011) or a linear combination of divergences (McClosky, 2010) can be successfully used to link the similarity between domains to the performance of a natural language processing system, no consensus exists about which divergence or combination of divergences is best suited for the task. The best divergence is not selected on theoretical grounds but by testing a range of divergences and selecting the best one. Although this is a valid working method, in this paper we investigate if it is possible to select the best measure for a given task using the correlation of the divergence with the performance.

3. Similarity measures

A text corpus needs to be converted into a measurable representation if the goal is to express similarity between two corpora by means of a single figure. Examples of such representations are: a single figure (e.g. the average sentence length in the corpus) or a distribution (e.g. the relative frequencies of the unique tokens in a corpus). Similarity can be expressed by the difference between two representations or between combinations of representations. In this paper, we use similarity measures that are based on distributions, which are simple, yet expressive, representations of a corpus. A distribution P can be described formally as:

$$P = \left\{ p_k : p_k \in \mathbb{R}^+ \wedge \sum_i^n p_i = 1 \right\} \tag{1}$$

with $k \in \mathbb{N}$, a unique identifier for each unique token (=type), with p_k the relative frequency of a type k in the corpus, and n the number of unique tokens in the text corpus. Based on these distributions, the following similarity measures are tested in this paper: Kullback-Leibler divergence, KL (Kullback & Leibler, 1951), Rényi divergence, R (Rényi, 1961), Skew divergence, S (Lee, 1999), Jensen-Shannon divergence, JS (Lin, 1991), simple Unknown Word Ratio, sUWR (Zhang & Wang, 2009), and overlap. Overlap is the conceptual complement to sUWR.

Given two distributions: P based on a test corpus T and Q based on a training corpus S, the formulas of the similarity measures are:

$$KL(P;Q) = \sum_k p_k log_2\Big(\frac{p_k}{q_k}\Big) \tag{2}$$

$$R(P;Q;\alpha) = \frac{1}{(\alpha-1)} log_2\Big(\sum_k p_k^\alpha q_k^{1-\alpha}\Big) \text{ with } \alpha \geq 0 \tag{3}$$

$$S(P;Q) = KL\big(Q;\alpha P + (1-\alpha)Q\big) \text{ with } \alpha \in [0,1] \tag{4}$$

$$JS(P;Q) = \frac{1}{2}\Big[KL\Big(P;\frac{P+Q}{2}\Big) + KL\Big(Q;\frac{P+Q}{2}\Big)\Big] \tag{5}$$

$$sUWR = \frac{|\{k : p_k \neq 0 \wedge q_k = 0\}|}{|\{k : p_k \neq 0\}|} \tag{6}$$

$$Overlap = \frac{|\{k : p_k = 0 \wedge q_k \neq 0\}|}{|\{k : q_k \neq 0\}|} \tag{7}$$

With p_k the relative frequency of type k in corpus P, q_k the relative frequency of type k in corpus Q. If a type is not present in a distribution, it adopts a relative probability of 0.[1]

The measures have been chosen based on their suitability in tasks such as parsing and part-of-speech tagging (Lee, 2001; Daumé III & Marcu, 2006; Zhang & Wang, 2009; Van Asch & Daelemans, 2010; Plank, 2011) and overlap is chosen because it is an unsuitable measure. Overlap measures the proportion of types present in the training corpus, but not included in the test corpus. It is clear that this information is not necessarily helpful for predicting accuracy. The purpose is to have a similarity measure that deviates from the others.

4. NLP machine learning task

4.1. British National Corpus

The corpus that is used for the experiments is the British National corpus, BNC (BNC, 2001). This corpus contains part-of-speech labels and is divided into different domains.

Table 1. Overview of number of tokens and sentences in each domain of the BNC.

DOMAIN	# TOKENS	# SENTENCES
IMAGINATIVE	19,507,596	1,333,450
WORLD AFFAIRS	17,925,728	726,881
SOCIAL SCIENCE	13,481,239	542,410
LEISURE	11,088,447	560,094
ARTS	7,182,257	303,019
APPLIED SCIENCE	7,154,185	312,948
COMMERCE & FINANCE	6,787,847	302,455
NATURAL & PURE SCIENCE	4,095,326	172,836
BELIEF & THOUGHT	3,160,642	136,366

The BNC annotators provided nine domain codes (*i.e.* wridom codes), making it possible to divide the text from books and periodicals into nine subcorpora. These annotated semantic domains are: imaginative (wridom1), natural & pure science (wridom2), applied science (wridom3), social science (wridom4), world affairs (wridom5), commerce & finance (wridom6), arts (wridom7), belief & thought (wridom8), and leisure (wridom9). The smallest domain is the belief & thought domain, consisting of ∼3M tokens, see Table 1. To eliminate the influence of different corpus sizes, a random selection of approximately 1,500,000 tokens has been taken from each domain. During sampling, sentences are kept intact.

[1]For the Kullback-Leibler divergence, if $p_k \neq 0$ but $q_k = 0$, smoothing is applied, such that $q_k = 2^{-52}$.

4.2. Part-of-speech tagging

In this paper, we have chosen the part-of-speech tagging machine learning task, because of the substantial influence of domain differences on the performance for this task. The machine learner that is used for the experiments is the memory-based part-of-speech-tagger, MBT (Daelemans & van den Bosch, 2005). MBT[2] is a machine learner that stores examples in memory and uses an extension of the kNN algorithm to assign part-of-speech labels. The default settings were used. An advantage of MBT is its speed, making it the machine learner of choice to carry out a high number of experiments. In addition, the conclusions of this paper do not hinge upon the choice of the machine learner, since the linear correlation between similarity measure and accuracy is observed for other machine learners (Van Asch, 2012).

5. Self-training setup

5.1. Procedure

Self-training is a technique consisting of automatically labeling additional training data in a semi-supervised way, before running an experiment (Charniak, 1997; McClosky, 2010; Sagae, 2010). Jiang and Zhai (2007) present an example for part-of-speech tagging.

Three corpora are needed for self-training: a labeled, training corpus, a labeled test corpus, and an unlabeled additional corpus. During self-training, a model is learned from the training data and it is applied to the unlabeled data. Thus, the additional training data is created by automatically labeling unlabeled data. Next, the (partially incorrectly) labeled, additional data is appended to the original training data (*self-training step 1*). This first labeling step is followed by a second training phase. The model resulting from this phase is then used to label the test data (*self-training step 2*).

It remains under debate whether self-training is a useful method; it is not shown to lead to performance gain in every experimental setup. Sagae (2010) argues that self-training is only beneficial in those situations where the training and test data are sufficiently dissimilar, but other factors – such as labeling accuracy of the unlabeled data – have an influence too. It would be helpful if the positive effect of the application of self-

[2]Available at http://ilk.uvt.nl/mbt (Last accessed: March 2013)

training could be determined in advance. Thus, given a set of three corpora, the experimental question is: *Does a given setup lead to an accuracy increase when self-training is applied?*

5.2. Evaluation and performance indicator

F-score[3] can be used for evaluating the setups (van Rijsbergen, 1975). A *true positive (tp)* is a three-corpus setup that results in an accuracy increase and that has been predicted to benefit from self-training. A *false positive (fp)* is a setup that does not benefit from self-training, although it was predicted to do so. A *false negative (fn)* is a setup that benefits from self-training, but was predicted the converse.

For the experiments of this paper, when each setup is predicted to lead to accuracy gain, the F-score would be only 25.61% (see Section 6.2). This baseline is an indication of the general success of self-training. If self-training would always be helpful, this baseline would be 100%. But since this is not the case, the low baseline is an incentive to look for a way to predict whether self-training will be increasing performance or not for a given combination of corpora. To this end, a performance indicator δ is designed.

In our design, the performance indicator is a binary indicator: If the performance indicator is positive for a given setup, self-training is considered to be beneficial. If the indicator is negative, no gain is to be expected.

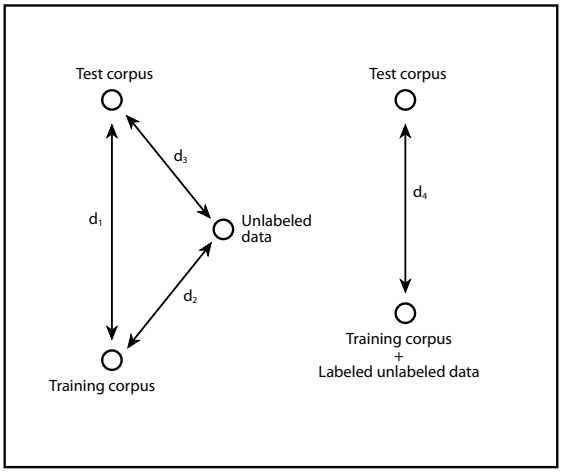


Figure 1. Theoretical justification of the performance indicator δ: Overview of similarities.

Figures 1 and 2 illustrate the rationale behind the design of the performance indicator δ. Figure 1 shows the different similarities that can be measured. d_1 represents the similarity between the training corpus and the test corpus. This is the only similarity involved when a straightforward test/train experiment is run. d_2 is the similarity between the training corpus and the additional unlabeled data. The labeling accuracy in the first self-training step is correlated with this similarity. d_3 is the similarity between the additional data and the test corpus. The more similar the test corpus and the additional data, the more beneficial self-training will be, provided that labeling during the first self-training step is near perfect. d_4 is the similarity between the composite corpus of training and additional data on the one side and the test data on the other. When labeling during the first self-training step would be perfect, the proportionality between d_4 and its associated accuracy (acc_4) would be the same as between d_1 and its accuracy (acc_1), since there would be no conceptual difference: both are measured between perfectly labeled corpora.

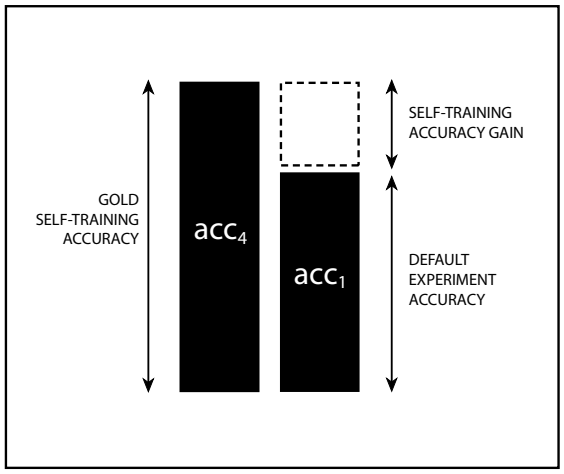


Figure 2. Theoretical justification of the performance indicator δ: Self-training accuracy gain.

It is known that $accuracy \propto \frac{1}{similarity}$ (Van Asch & Daelemans, 2010).[4] In a first step, the most important similarities are the similarity between *training corpus* and *test corpus* (d_1) and the similarity between *training corpus + additional data* and *test corpus* (d_4). The right column in Figure 2 depicts the accuracy of a regular test/train experiment (acc_1), and the height of this column is inverse proportional to the similarity d_1. Consider the case when labeling is perfect during the labeling step of a self-training experiment. In this case, the left column of Figure 2 is the highest obtain-

[3] F-score $= \frac{(1+\beta^2)\ tp}{(1+\beta^2)\ tp+\beta^2\ fp+fn}$; In this paper, β is set to 1.

[4] In this interpretation, the similarity value should be smaller when corpora are more alike.

able accuracy with self-training (acc_4). The perfectly labeled composite corpus serves as the training corpus. More data often leads to a higher performance e.g. Daelemans et al. (1999) and for that reason the left column is made higher than the right column.

The difference between acc_4 and acc_1 is the dashed column, which is the gain, obtained with (perfect) self-training, over a regular experiment. The performance indicator can be defined as

$$\delta' = \frac{acc_4}{acc_1} \tag{8}$$

if δ is larger than 1, self-training gain can be expected; if δ is smaller than 1, no gain is expected from self-training. Since we want to predict performance gain without running experiments, the accuracies are not available, but it is possible to use the similarities instead. In addition, the similarity between the *unlabeled data* and the *test data* (d_3) can be used as a proxy for d_4. Rewriting the performance indicator such that its outcome is binary then yields:

$$\delta = \frac{\left|\frac{d_1}{d_3} - 1\right|}{\frac{d_1}{d_3} - 1} \tag{9}$$

If δ is +1, gain is expected; if δ is -1, no gain is expected. The predictive power of this performance indicator is tested for part-of-speech self-training experiments in the next section.

6. Experiments

The corpus, the experimental setup, the evaluation method and the performance indicator have been presented in the previous section. In this section, these elements are used to conduct the experiments. First, the correlation coefficient for the different similarity measures is retrieved. Next, the self-training experiments are discussed.

6.1. Correlation r^2

The British national corpus contains nine domains, making it possible to select $\binom{9}{2} = 36$ different combinations of domains. The sets are used to conduct straightforward test/train experiments. Since it makes a difference whether a domain is selected as the first, i.e. as training corpus, or as the second, i.e. as test corpus, $36 \cdot 2! = 72$ experiments can be run.

By running the 72 part-of-speech tagging experiments, it is possible to compute the r^2 between the similarity measures between the test and training corpus on the one hand and the accuracy of the experiment on the other. In practice, each of the 72 experiments is a 25-fold cross-validation experiment. The training corpus is divided into five equal parts and the same is done for the test corpus. Next each training part is combined once with each test part in a part-of-speech tagging experiment with MBT. The final part-of-speech tagging accuracy and the similarity value are the averages of this cross-validation setup.

These experiments can be run while varying the similarity measure. The different correlations that are obtained in this manner will be used to differentiate the better from the worse similarity measures.

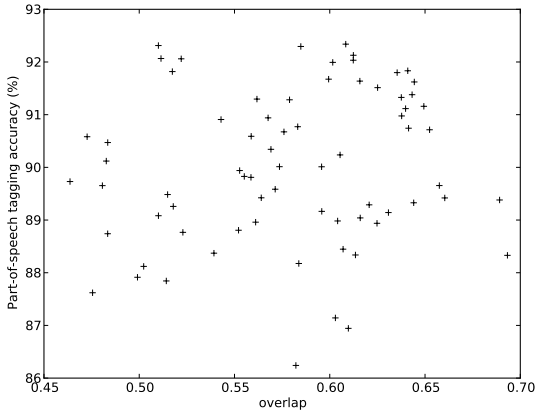
Table 2. The r^2 correlation coefficients for different similarity measures. The correlation is computed between similarity value and accuracy for 72 part-of-speech tagging experiments.

Measure	r^2
Rényi	0.083 – 0.987
Kullback-Leibler	0.986
Skew	0.224 – 0.985
sUWR	0.874
Jensen-Shannon	0.863
Overlap	0.051

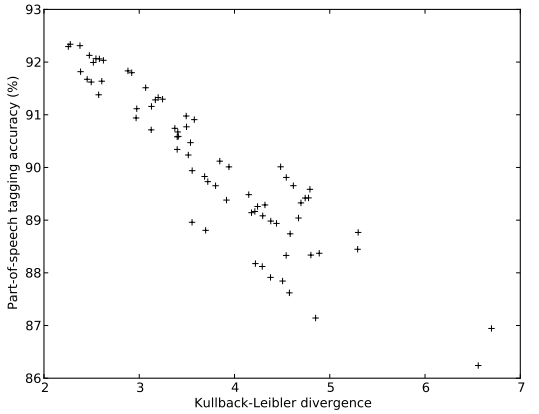
Table 2 shows the correlation coefficients r^2 for the selected set of similarity measures. Since the Skew and Rényi divergence contain a parameter α, a range of correlation coefficients is reported for these similarity measures.

The asymmetry of a measure M is the property that makes that the order of the distributions has an influence: $M(P,Q) \neq M(Q,P)$. Because all measures but the Jensen-Shannon divergence are asymmetric, the reported r^2 is an average value. For each run of 72 experiments, nine correlations are computed. One correlation for each set of 8 experiments for which the test corpus is the same. Averaging these values gives the values of Table 2. If all 72 experiments would be used to calculate a single overall r^2, the value for the Jensen-Shannon divergence would be too low since this measure cannot accommodate to the asymmetry of a part-of-speech tagging experiment: the similarity value is the same for $JS(P,Q)$ and $JS(Q,P)$, but the accuracy will be different. Splitting the computation of r^2 into a separate r^2 associated with each different test corpus, overcomes this incongruence, since $JS(P,Q)$ and $JS(Q,P)$ are no longer used for the calculation of the same r^2. Averaging all r^2's will aggregate the separate correlations to a single number.

As can be seen in Table 2, all measures show a good correlation except for overlap, which has been included for contrast. Two examples plots are given in Figure 3, along with the associated average r^2 value.



(a) Overlap ($r^2 = 0.051$)



(b) Kullback-Leibler ($r^2 = 0.986$)

Figure 3. Plot of two correlations between similarity value and part-of-speech accuracy for 72 experiments.

The parameterized divergences can also be adapted in such manner that they perform better or worse. The Rényi divergence has been applied with α varying from 0.02 to 0.98 in steps of 0.02. The higher α, the better the correlation. The Skew divergence with α values varying from 0.02 to 0.98 in steps of 0.02, varying from 0.9805 to 0.9995 in steps of 0.0005, and varying from 0.9995005 to 0.9999995 in steps of 5.10^{-7}. The higher α, the lower the correlation. Because the correlation of the Skew divergence declines much slower than the correlation of the Rényi divergence, more and smaller steps are computed for the Skew divergence.

6.2. Self-training gain prediction

The British National Corpus consists of nine domains and a set of three different corpora is needed to carry out a self-training experiment. This means that there are $\binom{9}{3} = 84$ possible sets. Since the order is important, and there are 3! permutations possible per set. In the end, this adds up to 504 experimental setups, using each domain either as training data, test data, or additional data.

As mentioned in Section 5.2, the baseline F-score when each self-training setup is expected to be beneficial is 25.61%. It should be stressed that a whole set of self-training setups are tested in this paper. As the baseline indicates, self-training may help performance, but it is not guaranteed. When examining a self-training setup for a single run of natural language processing task, one should be aware of the fact that a positive (or negative) outcome may be attributed to the corpora that have been selected. The single outcome should not give rise to conclusions about the general usability of the self-training technique for that task.

In this paper, when the 504 setups are tested during self-training experiments, only 74 experiments benefit from self-training. Leading to an F-score of $\frac{2 \cdot 74}{2 \cdot 74 + 430 + 0} = 25.61\%$.

When self-training is beneficial, the average performance gain is 0.07%, which amounts to an absolute difference of ∼985 tokens that are labeled correctly thanks to self-training. When self-training is harmful, the average performance loss is 0.09% or an extra of ∼1284 incorrectly labeled tokens. Overall, self-training has only a minor influence on accuracy, but even this minor influence can be predicted as is shown in the following experiments.

The derivation of the performance does not put severe constraints on the similarity measure that needs to be incorporated. The only requirements being that the value of the measure should never switch sign and that more similarity should lead to a smaller value. The 504 experiments are repeated while the similarity measure is replaced by one of similarity measures that are presented in Section 3.

We run a full round of experiments for the following similarity measures: Jensen-Shannon, Kullback-Leibler, sUWR, overlap, Rényi divergence with α varying from 0.02 to 0.98 in steps of 0.02, and Skew divergence with α varying from 0.02 to 0.98 in steps of 0.02, varying from 0.9805 to 0.9995 in steps of 0.0005, and varying from 0.9995005 to 0.9999995 in steps of

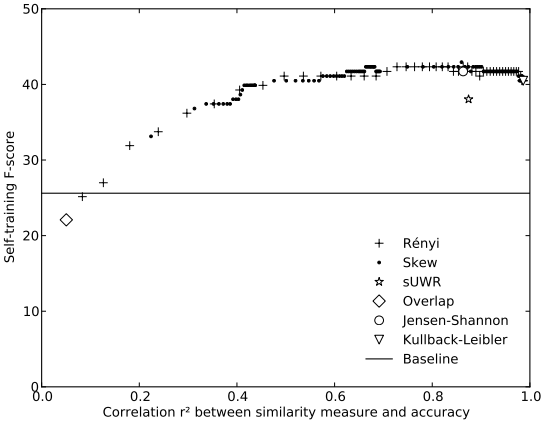


Figure 4. The variance of the self-training F-score for similarity measures that show different r^2 correlations for the accuracy of straightforward part-of-speech labeling experiments and the degree of test/train-similarity as expressed by that measure. Each point is a different measure. For the parameterized measures (Rényi and Skew), each point is the measure with a different α value, $\alpha \in]0, 1[$. For the Rényi divergence, an increasing α leads to a higher r^2. For the Skew divergence, an increasing α leads to a lower r^2.

0.0000005.[5] The F-scores of these experiments are plotted in Figure 4 and the F-scores are given in Table 3. The y-axis indicates the F-score, the x-axis indicates the correlation of the used measure, as reported in Table 2.

Table 3. F-scores for different similarity measures when used in the performance indicator δ. Statistical difference with baseline is indicated with *.

Measure	F-score
Rényi	25.15 – 42.33*
Kullback-Leibler	40.49*
Skew	33.13* – 42.94*
sUWR	38.04*
Jensen-Shannon	41.72*
Overlap	22.09

A first conclusion that can be drawn from Table 3 and Figure 4 is that it is almost always better to use the performance indicator to predict whether a self-training setup will be beneficial than to assume that self-training is beneficial. Only the two similarity measures at the left of the figure fall below the previously reported baseline of 25.61%. These two are overlap ($r^2 = 0.051$) and Rényi with $\alpha = 0.02$ ($r^2 = 0.083$).

The second conclusion is that, in general, similarity measures that are better correlated with accuracy (higher r^2) are more suited to be used as the core of the performance indicator δ. Although this observation holds in general, Figure 4 also shows that there is a broad range of similarity measures that approach the maximum F-score, provided that a certain degree of correlation has been reached. It is even the case that prediction appears to be less trustworthy when the higher r^2 scores are reached. The best Skew divergence is with $\alpha = 0.82$, associated with an r^2 value of 0.861 and reaching an F-score of 42.94%. Although a feeble downward trend for the top r^2 values can be observed, there is no statistical difference[6] between e.g. sUWR and Jensen-Shannon ($p = 0.099$). Only a larger difference, like between Jensen-Shannon and overlap ($p = 9.10^{-6}$), is statistically significant.

A higher r^2 does not necessitate obtaining a higher F-score. This fact can also be illustrated when straightforward accuracy is used as the similarity measure. As mentioned before, the best way to predict the accuracy of an experiment is by running that experiment. We can derive a similarity measure from the accuracy of an experiment: $similarity\ value = \frac{1}{accuracy}$. It is clear that the correlation r^2, computed as in Table 2, for this measure is 1. This *perfect* similarity measure can now be used in the performance indicator δ. The associated F-score for self-training then becomes 40.49%, which is not markedly better than using any other efficient measure. Since there is no better similarity measure available, this figure can be considered as a limitation to the method of using correlation r^2 as the selection criterion for selecting the best measure for this task. Indeed, if r^2 would be the only factor into play, the F-score when accuracy is used as similarity measure should be highest. But this is not the case.

This observation has consequences on two levels. First, r^2 cannot be used as the single selection criterion for selecting the best measure to be used in the performance indicator, although a minimal r^2 value is required. Second, the design of the performance indicator may not be flexible enough to anticipate certain situations, such as a very unsuccessful first labeling step. This implies that, even if you have built in the best similarity measure, it remains impossible to correctly predict all experimental setups for which self-training

[5]Because of the large amount of data points for the Skew divergence, with relatively small correlation differences, not all points are shown in Figure 4.

[6]Stratified approximate randomization testing of F-score of the positive class has been used to assess the significance of different labeling scores of the test set (Noreen, 1989). Implementation: www.clips.ua.ac.be/scripts/art

is beneficial.

6.3. Influence of the α parameter

When examining the definitions of the Rényi and Skew divergence, eqs. (3) and (4), we can draw the following conclusions on the influence of the α parameter on the measure: For the Rényi divergence, it can be seen that lowering α implies lowering the influence of the test corpus (p_k^α becomes smaller, and $q_k^{1-\alpha}$ becomes larger). As can be seen in Figure 4 by moving from right to left, lowering the influence of the test corpus leads to a deteriorated performance of the similarity measure, after a small initial gain.

For the Skew divergence, lowering alpha also means lowering the influence of the test corpus. Moving from left to right in Figure 4, in the beginning, lowering the influence of the test corpus improves the performance of the similarity measure, but when an α value of 0.82 is reached, the best parameter setting is reached. Further lowering of the influence of the test corpus will eventually lead to performance decrease.

Conclusion and perspectives

In this paper we investigated the possibility to rank similarity measures according to appropriateness for self-training. Our approach offers an analytical and systematic method to select the best-suited similarity measure from a set of measures. This, in contrast to the more frequent practical approach of testing all similarity measures in order to find the measure fit for the task. An additional advantage of the framework is that it enables the investigation of other properties besides linear correlation. This may be a stimulus for further research focusing on objective ways to express domain differences between corpora.

The machine learning task we implemented, is a self-training part-of-speech tagging task, in which a similarity measure is used to obtain a prediction about the usefulness of the self-training setup. To predict the usefulness, a performance indicator δ has been designed. We found that the r^2 of a similarity measure can be used as a coarse selection criterion for selecting a set of suitable measures.

The fact that the correlation cannot be used to single out one best-suited measure can be attributed to two interfering causes. The first cause being that the correlation coefficient may disregard certain influential properties of similarity measures. The sensitivity to relative frequency differences or the interdependencies between tokens may be two of such undetected properties. A second cause making the correlation appear an insufficient selection criterion may be that the effectiveness of the performance indicator δ can be limited by its design. This last conclusion is corroborated by the observation that incorporating a *perfect* similarity measure (accuracy) does not lead to the best performance.

For parameterized similarity measures (Rényi and Skew divergence), we found that moderately lowering the influence of the test corpus in the measure leads to an increased performance. This observation may contribute to the design of parameterized variants of existing similarity measures (like e.g. a parameterized sUWR). The newly introduced parameter should regulate the proportional influence of test and training corpus.

In general, we can conclude that the use of similarity measures in natural language processing is mainly a trial-end-error approach. We made start at looking into the various properties of similarity measures by investigating the information carried by the correlation coefficient. But, as our research showed, other properties exist and following research could focus on conceiving new measures that can express these properties in an objective manner.

Acknowledgements

This research is funded by the Research Foundation Flanders (FWO-project G.0478.10 – Statistical Relational Learning of Natural Language) and made possible through financial support from the University of Antwerp (GOA project BIOGRAPH).

References

Biber, D., & Gray, B. (2010). Challenging stereotypes about academic writing: Complexity, elaboration, explicitness. *Journal of English for Academic Purposes*, *9*, 2–20.

BNC (2001). The British National Corpus, version 2 (BNC world). Available at http://www.natcorp.ox.ac.uk (Last accessed: March 2013).

Charniak, E. (1997). Statistical parsing with a context-free grammar and word statistics. *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Applications of Artificial Intelligence Conference* (pp. 598–603). Rhode Island, USA: MIT Press.

Chen, B., Lam, W., Tsang, I., & Wong, T.-L. (2009). Extracting discriminative concepts for domain adap-

tation in text mining. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 179–188). Paris, France: ACM.

Daelemans, W., & van den Bosch, A. (2005). *Memory-based language processing*. Studies in Natural Language Processing. Cambridge: Cambridge University Press.

Daelemans, W., Van Den Bosch, A., & Zavrel, J. (1999). Forgetting exceptions is harmful in language learning. *Machine Learning*, *34*, 11–41.

Daumé III, H., & Marcu, D. (2006). Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, *26*, 101–126.

Della Pietra, S., Della Pietra, V., & Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*, 380–393.

Gao, J., Goodman, J., Li, M., & Lee, K.-F. (2002). Toward a unified approach to statistical language modeling for chinese. *Transactions on Asian Language Information Processing*, *1*, 3–33.

Jiang, J., & Zhai, C. (2007). Instance weighting for domain adaptation in NLP. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* (pp. 264–271). Prague, Czech Republic: Association for Computational Linguistics.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, *22*, 79–86.

Lee, L. (1999). Measures of distributional similarity. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics* (pp. 25–32). Maryland, USA: Association for Computational Linguistics.

Lee, L. (2001). On the effectiveness of the skew divergence for statistical language analysis. 8^{th} *International Workshop on Artificial Intelligence and Statistics (AISTATS 2001)* (pp. 65–72). Florida, USA. Online repository http://www.gatsby.ucl.ac.uk/aistats/aistats2001 (Last accessed: March 2013).

Lin, J. (1991). Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, *37*, 145–151.

Mansour, Y., Mohri, M., & Rostamizadeh, A. (2009). Multiple source adaptation and the Rényi divergence. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (pp. 367–374). Montreal, Quebec, Canada: AUAI Press.

McClosky, D. (2010). *Any domain parsing: Automatic domain adaptation for natural language parsing*. Doctoral dissertation, Department of Computer Science, Brown University, Rhode Island, USA.

Moore, R. C., & Lewis, W. (2010). Intelligent selection of language model training data. *Proceedings of the ACL 2010 Conference Short Papers* (pp. 220–224). Uppsala, Sweden: Association for Computational Linguistics.

Noreen, E. W. (1989). *Computer-intensive methods for testing hypotheses*. New York, NY, USA: John Wiley.

Pearson, K. (1896). Mathematical contributions to the theory of evolution. – III. Regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London (Series A)*, *187*, 253–318.

Plank, B. (2011). *Domain adaptation for parsing*. Doctoral dissertation, University of Groningen, the Netherlands. Groningen Dissertations in Linguistics 96.

Ravi, S., Knight, K., & Soricut, R. (2008). Automatic prediction of parser accuracy. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing* (pp. 887–896). Honolulu, Hawaii: Association for Computational Linguistics.

Rényi, A. (1961). On measures of information and entropy. *Proceedings of the* 4^{th} *Berkeley Symposium on Mathematics, Statistics and Probability* (pp. 547–561). Berkeley, California, USA: University of California Press.

Sagae, K. (2010). Self-training without reranking for parser domain adaptation and its impact on semantic role labeling. *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing* (pp. 37–44). Uppsala, Sweden: Association for Computational Linguistics.

Van Asch, V. (2012). *Domain similarity measures: On the use of distance metrics in natural language processing*. Doctoral dissertation, University of Antwerp. Available at http://www.clips.ua.ac.be/bibliography/domain-similarity-measures.

Van Asch, V., & Daelemans, W. (2010). Using domain similarity for performance estimation. *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing* (pp. 31–36). Uppsala, Sweden: Association for Computational Linguistics.

van Rijsbergen, C. J. (1975). *Information retrieval*. London, UK: Butterworths.

Verspoor, K., Cohen, K. B., & Hunter, L. (2009). The textual characteristics of traditional and open access scientific journals are similar. *BMC Bioinformatics*, *10*, 1–16.

Zhang, Y., & Wang, R. (2009). Correlating natural language parser performance with statistical measures of the text. *Proceedings of the 32nd annual German conference on Advances in artificial intelligence* (pp. 217–224). Paderborn, Germany: Springer-Verlag.

Unsupervised identification of compounds

Suzanne Aussems S.H.J.A.AUSSEMS@TILBURGUNIVERSITY.EDU
Bas Goris B.C.C.GORIS@TILBURGUNIVERSITY.EDU
Vincent Lichtenberg V.J.J.LICHTENBERG@TILBURGUNIVERSITY.EDU
Nanne van Noord NANNE@TILBURGUNIVERSITY.EDU
Rick Smetsers R.H.A.M.SMETSERS@TILBURGUNIVERSITY.EDU
Menno van Zaanen MVZAANEN@TILBURGUNIVERSITY.EDU
Tilburg University, Tilburg, The Netherlands

Keywords: automatic compound recognition, point-wise mutual information, edge detection, unsupervised machine learning

Abstract

In a variety of languages, compounds (i.e., lexemes consisting of more than one stem) are written as one token. Identifying such compounds together with their compound boundaries can help improve the quality of computational linguistic tasks such as machine translation and spelling correction. In order to create annotated compound datasets, we need to be able to identify compounds in various languages. Since manual identification is very time consuming, we propose novel, language-independent approaches to the identification of compounds in sets of words. A range of methods has been explored, including unsupervised machine learning approaches. The most successful approach focuses on the identification of compound boundaries by identifying irregularities in letter combinations, exploiting point-wise mutual information values between letter n-grams. The results of applying of our methods to a collection of Dutch words show major improvements over a word-based compound identifier.

1. Introduction

In languages such as German, Russian, Finnish, Icelandic, Afrikaans and Dutch, the process of compounding is highly productive. New compounds can be created by combining two or more simplex words into a new word (van Huyssteen & van Zaanen, 2004). The meaning of the compound depends on the meaning of its parts. For instance, in Dutch, the compound *slakom* '*lettuce bowl*' consists of the simplex words *sla* '*lettuce*' and *kom* '*bowl*'. The process of compounding can be repeated, leading to compounds that have more than two simplex components, such as *slakomverkoper* '*lettuce bowl vendor*'.

Due to the productive nature of the process of compounding, fixed word lists will always be a limiting factor in describing the dictionaries of languages that allow for compounding. Automatic identification of compounds and compound boundaries can help improve upon the quality of linguistic applications such as machine translators and spelling checkers. For instance, being able to identify the parts of a compound allows for the translation of the parts (for instance, if *fietspad* '*bicycle path*' is not in the dictionary, but the compound boundary (**fiets + pad**) and the simplex words *fiets* '*bicycle*' and *pad* '*path*' are known, the word may still be translated). Similarly, spelling correctors require knowledge of the process of compounding to accept valid compounds.

In certain cases, the concatenation of simplex words into compounds requires a form of "glue". For instance, *instellingenmenu* '*setup menu*' contains the simplex words *instelling* '*setup*' and *menu* '*menu*' with the morpheme *en* serving as a glue. Such morphemes are called *linking morphemes* (Booij, 1996; Wiese, 1996).

Note that in other languages, such as English, components of compounds are written as separate tokens. The identification of such multi-word compound units

Appearing in *Proceedings of BENELEARN 2013*.

is not trivial (Mima & Ananiadou, 2000). While there are certain similarities between multi-word compound units and single token compounds, the differences are to such an extent that they require a different approach (Ryder, 1994).

Previous research concerned with splitting compounds has been conducted primarily in the context of machine translation (Koehn & Knight, 2003; Garera & Yarowsky, 2008; Macherey et al., 2011; Alfonseca et al., 2008). In previous studies, monolingual and bilingual approaches can be distinguished.

Several bilingual approaches have successfully used information from parallel corpora to identify compound boundaries. Koehn and Knight (2003) break up German compounds so that a one-to-one correspondence to English content words can be established. Part-of-speech tags and frequency information are used to align German compound parts to their English translations. Similarly, Macherey et al. (2011) apply dynamic programming to find one-to-one mappings to English translations of German and Greek compound parts. The emphasis on finding correct translations for compounds is also evident in Garera and Yarowsky (2008). While their approach does not require bilingual training text, the authors use cross language compound evidence obtained from bilingual dictionaries; an approach similar to Koehn and Knight (2003).

The purpose of these bilingual approaches is to improve the quality of machine translation systems. For their tasks, parallel corpora augmented with metadata are readily available. Monolingual methods aim to identify compound boundaries without the need of such information. Alfonseca et al. (2008) apply frequency and probability based methods, and search for compound components that occur in different anchor texts pointing to the same document. The combined features are used in a supervised machine learning classifier.

Research has been conducted on the segmentation of words in morphemes. The *Morfessor* system (Creutz & Lagus, 2005) implements an unsupervised method for producing a segmentation of morphemes. Its task is to model a language that consists of a lexicon of morphemes, by looking at high-frequency n-grams in unannotated corpus text. Segmentation is applied by matching morphemes in the constructed lexicon to words. As such, its approach is similar to our compound component approach, described in Section 3.1. Authors report precision scores as high as 63% and recall as high as 75% for English word type boundaries, and precision of 85% and recall of 53% for Finnish word type boundaries.

Our research is different from previous work in several regards. In the first place, our task is different as we aim to identify compounds and not compound boundaries. To achieve this, we investigate several unsupervised methods. In the second place, our monolingual data consists solely of individual lemmas, without additional frequency data, POS tags or other meta-information. As such, our methods can be applied to identify compounds in languages where such information is not readily available.

The task of identifying compounds in non-trivial. There are even cases in which it is unclear whether a word is a compound (without further context). For instance, the word *minister* can mean '*minister*' or '*mini star*'. In the first case, the word is a simplex word, whereas in the second situation, the word is a compound **mini + ster**. Such situations happen regularly as certain affixes can also serve as simplex words. For instance, *vergeven* '*to forgive*' is not a compound, but can be analyzed as one **ver + geven** '*far to give*'. Another such affix is *-lijk*, for instance in *mannelijk* '*manly*', but the simplex word *lijk* translates to '*corpse*'.

van Huyssteen and van Zaanen (2004) propose two compound analysis systems for compounds in Afrikaans, that recognize compound boundaries and linking morphemes if present. The first system is based on a longest string-matching algorithm that searches for known (simplex) words at the beginning and at the end of a potential compound. If the analysis of a potential compound shows that it consists of valid and shorter words from a word list, possibly glued together using valid linking morphemes, the word is considered a compound. This algorithm forms the basis of the compound component identifier as described in Section 3.1.

The second system is an unsupervised clustering approach that, based on k-means, groups words in either a compound or non-compound class. The clustering is performed based on shallow, word-driven features.

The final system is an unsupervised machine learning classification approach that decides whether there is a compound boundary or linking morpheme boundary between any of the letters in the word. Using a sliding window, all positions between letters in the word are considered. The letters left and right of the between-letter positions are used as features in the machine learning classifier. The systems described in this paper are unsupervised and as such do not require any annotated data.

2. Problem description

Ultimately, we would like to be able to develop systems that automatically identify compound and linking morpheme boundaries given a compound. However, developing and evaluating such systems requires datasets annotated with compound boundaries.

Unfortunately, only a very limited number of annotated compounds datasets exist. For instance, the CKarma dataset (CText, 2005; Pilon & Puttkammer, 2008) contains 72,849 annotated Afrikaans compounds and for Dutch, the e-Lex dataset (see Section 4.1) is available, containing 88,713 annotated compounds.

To create larger datasets of (manually) annotated compounds, or to create such datasets for languages for which no such datasets yet exist, access to a dataset containing (only) compounds is required. The research described here aims to tackle the problem of automatically identifying compounds in large plain text corpora.

Following the identification of compounds, the annotation task is to manually identify the compound and linking morpheme boundaries. Manually annotating such a list of types is an expertise, time and resource intensive task as each word has to be considered.

Systems that identify compounds in corpora should aim for the identification of as many compounds as possible (high recall), while limiting the incorporation of non-compounds as much as possible (high precision). Compounds that are not identified by the system will never be considered for manual annotation at all, but non-compounds can still be discarded during the compound boundary annotation.

3. System description

We have experimented with a variety of systems to identify compounds given a set of words. First, we describe the compound component identifier, which is a word-based approach. Second, we propose an unsupervised clustering approach, which aims to identify compounds based on a combination of shallow features. Finally, we describe approaches that aim to identify potential compound boundaries based on the pointwise mutual information values between letters, which indicate the regularity of letters occurring next to each other. Letter combinations that do not co-locate regularly are likely to be compound boundaries.

The systems described here receive only a set of types as input. No frequency information is given, because often no reliable counts can be obtained. For instance, for Afrikaans, no publicly available large scale corpus exists. In case a system requires additional information, this is indicated in the description of the system. Ideally, the developed systems are completely unsupervised and language independent.

3.1. Compound component identifier

Based on the idea that compounds consist of two or more meaningful lexical stems, van Huyssteen and van Zaanen (2004) developed a system, called *longest string-matching*, that identifies compound boundaries in compounds. The algorithm takes a compound as input and recursively searches a dictionary (containing simplex words) for words that can be concatenated to form a compound. The system is also able to deal with linking morphemes, which are provided beforehand.

The compound component (CC) identifier follows the idea of the longest string-matching approach closely. However, in contrast to the longest string-matching algorithm, the CC identifier does not have access to a list of simplex words. The CC identifier solves this by making use of all the words in the input dataset. To be more precise, the CC identifier receives a comprehensive set of input data containing both simplex and compound types. The CC identifier then searches for types that can be constructed by concatenating two or more other types from the set. If the CC identifies such a construction, the system classifies it as a compound.

The CC identifier is provided with a list of valid linking morphemes beforehand. This information is language dependent, so the system requires minor supervision by an expert. We have taken into account that very short simplex words lead to over-generation of the system (identifying too many words as compounds). Specifically two letter words, which are typically function words, such as determiners, prepositions or conjunctions, (for instance *in* '*in*', *op* '*on*', *na* '*after*', or *en* '*and*') have a large negative impact. Despite being valid simplex words, they can be used incorrectly as a stem in a compound. For instance, the non-compound *inbedden* '*to embed*' can be split into **in** + **bedden** '*in* + *beds*'. To solve this problem, we limit the minimum component length to three letters.

3.2. k-Means identifier

The idea behind the k-means identifier (KM) is that a combination of surface properties of words may be enough to identify words as compounds. For instance, because compounds are constructed using several simplex words, on average they tend to be longer than simplex words or contain more syllables. Obviously, each feature by itself does not provide enough infor-

mation to decide whether a word is a compound or not, but a combination of features may yield useful results.

The input of the KM identifier is, like the other systems we describe here, a set of types. Shallow features are extracted, leading to a feature vector for each type. The collection of feature vectors serves as the input of a k-means clustering algorithm. The value of k describes the number of classes, which is set manually beforehand. All possible combinations of the following features have been used in the experiments.

word length the number of characters in the word;

longest vowel cluster length the length of the longest sequence of adjacent vowels in the word;

vowel cluster count the number of vowel clusters in the word (serving as a rough approximation of syllable count);

cumulative bi-gram probability the sum of the letter bi-gram log probabilities of the word (n-gram probabilities are always smaller than 1, so we take the absolute values of the log probabilities);

cumulative tri-gram probability the sum of the absolute values of the letter tri-gram log probabilities in the word;

cumulative 4-gram probability the sum of the absolute values of the letter 4-gram log probabilities in the word;

cumulative 5-gram probability the sum of the absolute values of the letter 5-gram log probabilities in the word;

valid word regex classification boolean classification based on a regular expression match that yields false if a substring of the word consists of a sequence of three non-alphabetic characters, otherwise the word is classified as true. This feature was added to reduce noise and the influence of digit combinations in datasets.

3.3. Point-wise mutual information identifier

The approach we describe in this section is based on the idea that compound boundaries can be found where "unusual" letter combinations occur. The underlying idea is based on the following observed properties of language:

1. the unequal distribution of character n-grams found in languages such as English and Chinese (Ha et al., 2003);

2. our observation that in simplex and non-compound words certain letter combinations occur more often in isolation than together.

These properties may be different between simplex words and compounds, since compounds contain letter combinations that disregard the unequal distribution of letter sequences (specifically around the boundary of the simplex words in the compound). To clarify, a compound can consist of two simplex words each by itself adheres to the non-linear distribution of letter combinations (n-grams). However, the compound boundary does not, because compound boundary letter combinations simply consist of the beginning and ending of the simplex components.

To measure the regularity of letter combinations, we use the point-wise mutual information (PMI) metric. The following paragraphs explain the process of calculating these values, but let us exemplify its use first with the following example. The word *boekenbeurs* '*book convention*' has a compound boundary between *boeken* and *beurs*.[1] Figure 1 shows that the letter combinations on compound boundary *enbe*, *en* and *be*, occur more often in isolation than together. In other words, the PMI value for the two bi-grams is low, which shows that it is surprising that these characters occur together.

The calculation of the point-wise mutual information (PMI) metric between two letter n-grams, A and B is calculated according to Equation 1. The probability of the concatenated n-grams (AB) is divided by the product of the probabilities of the n-grams by themselves. The results described here are all generated using $n = 2$. Note that the results of the fraction is always smaller than or equal to 1. Taking the log leads to a negative result where smaller fractions lead to larger PMI values. The calculation of the probability of a letter n-gram is computed using the relative frequency (see Equation 2).

$$pmi(A, B) = \log \frac{P(AB)}{P(A) * P(B)} \quad (1)$$

$$P(X) = \frac{\text{count}(X)}{|\text{corpus}|} \quad (2)$$

[1] The simplex *boeken* contains a linking morpheme boundary, but this is irrelevant when identifying compounds.

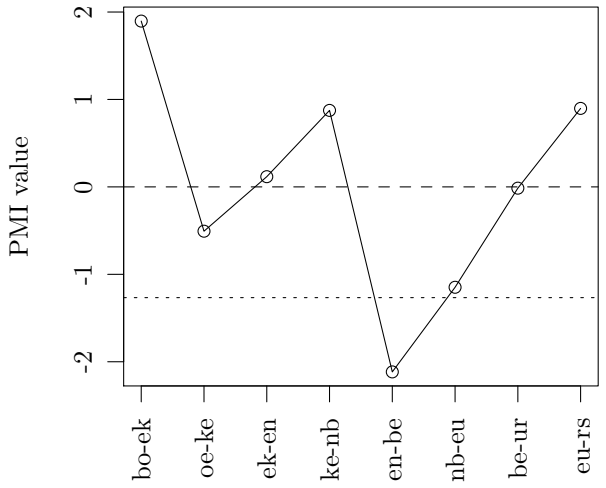


Figure 1. PMI values for the boundaries in *boekenbeurs* '*book convention*'. The dashed line is the mean, the dotted line is one standard deviation below the mean.

Given a word, PMI values are computed by applying a sliding window over the word. Equation 1 is applied to each letter $n*2$-gram, where the relative frequency of the two adjacent letter n-grams constitute A and B respectively. For a word of length k, this procedure yields a set of $k-(n*2)$ PMI values (see Equation 3). Here pmi_0 represents the application of the *pmi* function on the first letter $n*2$-gram in the word. For example, PMI for letter bi-grams in *fietspad* '*bicycle path*' yields a set of values for letter 4-grams: *fiet*, *iets*, *etsp*, *tspa* and *spad*.

$$PMI = \{pmi_i\}_{i=0}^{k-(n*2)} \tag{3}$$

After calculating PMI the resulting values are standardized according to Equation 4, where μ is the mean of the values and σ is the standard deviation. The standardized PMI values (z) are evaluated on their distance to μ. If a value exceeds a given threshold parameter t, which signifies the number of standard deviations from the mean, a word is identified as being a compound.

$$z = \frac{x-\mu}{\sigma} \tag{4}$$

3.4. PMI edge detection identifier

To improve compound identification performance, edge detection filters are applied to PMI values obtained from the system described in Section 3.3. Edge detection is a technique widely used in the field of signal processing to detect discontinuities in multi-dimensional signals. A small matrix, called a *kernel*, is applied to the signal, to either sharpen or smoothen the input. This is accomplished by means of convolution between the kernel and the input signal (Ziou & Tabbone, 1998). Since our data is of a one-dimensional nature, we utilize vector kernels instead of matrices. It is expected that compound boundaries (i.e., dips in PMI values) are more easily detectable after application of an edge detection kernel.

Kernel convolution usually requires values from outside the range of the PMI values. Therefore, "edges" (i.e., the first and last PMI values) are extended with length k, which is calculated according to Equation 5. Then, a sliding window is applied and dot products of the kernel and the PMI values are calculated according to Equation 6. Resulting values are standardized according to Equation 4 and evaluated on the number of standard deviations they differ from the mean. As such, evaluation is identical to that of "regular" PMI values, described in Section 3.3.

$$k = \lfloor \frac{|\text{kernel}|}{2} \rfloor \tag{5}$$

$$\text{PMI}_{\text{filtered}} = \left\{ \sum_{i=0}^{|\text{kernel}|} \text{pmi}_{j+(i-k)} \text{kernel}_i \right\}_{j=0}^{|\text{PMI}|} \tag{6}$$

Two types of convolution kernels are evaluated here, based on *Gaussian* (Equation 7) and *sigmoid* (Equation 8) functions respectively. Kernel values are calculated based on a template and an input parameter p. To ensure that the output values are of the same relative magnitude as the input values, the kernel values are normalized so that the sum of their values equals 1. All kernels we evaluated are of length three (yielding context $k = \lfloor \frac{3}{2} \rfloor = 1$).

$$\text{kernel}_{\text{Gaussian},p} = \left\{ \frac{1}{3} - p, \frac{1}{3} + 2p, \frac{1}{3} - p \right\} \tag{7}$$

$$\text{kernel}_{\text{sigmoid},p} = \left\{ \frac{1}{3} - p, \frac{1}{3}, \frac{1}{3} + p \right\} \tag{8}$$

As shown in Figure 2 (compared to Figure 1) differences between PMI values get larger when edge-detection is applied to the PMI of the word *boekenbeurs*. It is expected that edge detection techniques

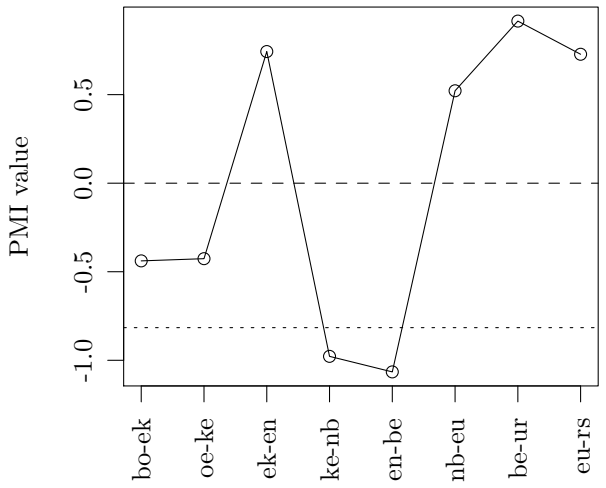


Figure 2. Sigmoid edge detection with $p = 1$ applied to PMI values for the boundaries of the word *boekenbeurs*. The dashed line is the mean, the dotted line is one standard deviation below the mean.

amplify compound boundaries and help identifying compounds.

4. Results

4.1. Dataset

All systems are tested on the Dutch e-Lex dataset (`http://tst-centrale.org/nl/producten/lexica/e-lex/7-25`), which contains a total of 96,219 morphologically segmented lemma entries. The morphological segmentations of the lemmata were used during evaluation of our systems to determine which words were compounds. We identified compounds by parsing hierarchical morphological segmentations. A total of 68,686 lemma entries that contain compositional morphology were identified as compounds. The remaining 27,533 words are simplex words possibly annotated with derivational morphology structure.

4.2. System results

The results of all systems can be found in Table 1. The compound component identifier has the highest precision, which was expected as it exploits the fact that compounds are built from simplex words and e-Lex contains a substantial amount of simplex words. The CC identifier does not have any manually tunable parameters. As such, it can not be modified to improve recall.

Table 1. Results for all systems on the e-Lex dataset. P is precision, R is recall and F_1 is F_1 score. For k-means experiments, feature combinations for optimal F_β scores are selected. For the PMI experiments, N, G and S indicates that no edge detection, Gaussian or sigmoid filters have been applied, p indicates the kernel parameter value and t refers to the threshold used for classification. Italics indicate system dependent best results. Overall best results are shown in boldface.

	P	R	F_1
Compound component identifier			
	89.603	*63.268*	*74.166*
k-means identifier			
max $F_{0.5}$, $k = 2$	*83.842*	61.573	71.002
max F_1, $k = 2$	76.236	*80.134*	*78.136*
max F_2, $k = 2$	76.236	*80.134*	*78.136*
Point-wise mutual information identifier			
N, $t = 1$	*74.683*	73.842	74.260
N, $t = 0.1$	73.092	**99.999**	84.454
N, $t = 0.3$	73.103	99.987	*84.457*
G, $p = 0.3, t = 1$	*74.754*	78.051	76.367
G, $p = 0.5, t = 0.2$	73.096	**99.999**	84.456
G, $p = 0.7, t = 0.2$	73.118	99.975	**84.463**
S, $p = 0.3, t = 1$	*73.816*	88.989	80.695
S, $p = -1.9, t = 0.1$	73.093	**99.999**	*84.455*

The k-means identifier is applied to the data with $k = 2$ target classes. In each experiment, the class that fits the compound class best, is selected as the compound class. The best results are found when using only a sub-set of the features. When optimizing on $F_{0.5}$, the longest vowel cluster length and cumulative 5-gram probability features are used. When optimizing F_1 or F_2, only the vowel cluster count feature is used.

In an attempt to improve precision and recall values separately, experiments have been conducted by maximizing commonly used F_β measures. The $F_{0.5}$ measure weights precision higher than recall and the F_2 measure puts more emphasis on recall than precision as shown in Equation 9. It is interesting to note that the $F_{0.5}$ optimized k-means identifier is able to produce results comparable to the compound component identifier, even though the sources of information are completely different.

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \qquad (9)$$

The threshold t for evaluating PMI values is set to values between 0.1 and 3.5 standard deviations from the mean. The best F_1 and recall scores for all three PMI variants (unfiltered, sigmoid and Gaussian) are found

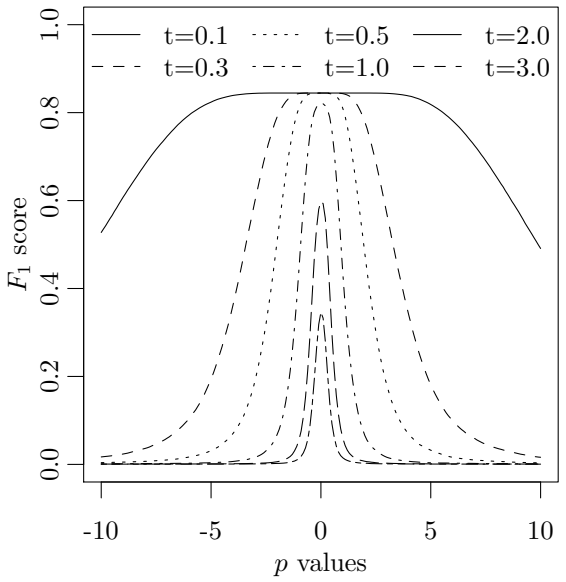


Figure 3. F_1 score results for the sigmoid filter with varying t and p values.

with t values between 0.1 and 0.3. The best results of all three systems are comparable. Best precision is obtained with slightly higher thresholds, ranging between 0.5 and 1. Increasing the threshold to values higher than 1 rapidly degrades performance: unfiltered PMI values evaluated to a threshold of 1.5 already result in a F_1 score of 45.570, while a threshold of 3.5 only yields a F_1 score of 1.744, which is similar to the Gaussian and sigmoid results.

Precision obtained by PMI variants never reach those generated by the compound component and k-means methods.

Even though the Gaussian, sigmoid and unfiltered PMI settings all yield significantly different distributions of F_1 scores , their behavior is similar when varying the values of the parameter p (varying the impact of the values in the context). Both increasing or decreasing p to a certain extent lead to similar precision, recall, and F_1 scores. The best F_1 scores are obtained with p around 0. Performance rapidly decreases with larger absolute p values. This is illustrated in Figure 3. Here we see the F_1 score results of the sigmoid filter with various values for t and p. This indicates that essentially the application of the kernels does not lead to improved results.

5. Conclusion

In this paper we have investigated different approaches to the identification of compounds in a set of words. These systems will be used to select compounds that will be structurally annotated for compound boundaries by human annotators in a later stage.

The compound component (CC) system is based on a similar system that has been used in the past to identify compound boundaries automatically. The CC system identifies compounds in a set of words by identifying components (which are simplex words) in the given set of words. This system leads to a high precision. A second approach, based on the unsupervised k-means clustering using shallow features of words leads to similar performance using different information.

We compared these results against a third approach, which is based on point-wise mutual information (PMI) values of the consecutive letter combinations in the word. These systems result in the highest recall (near 100%) and F_1 scores (over 84%).

As future work, we are interested in using the PMI system to identify the actual compound boundaries (similarly to the use of the original CC system and others described as previous research earlier). However, preliminary results show that the PMI systems also identify boundaries that are not actual compound boundaries, but may correspond to other morphological boundaries. In the setting described here, this is not a problem. A word is considered a compound if *at least* one boundary is found. However, identifying too many boundaries has a negative effect on the precision of the identification of compound boundaries. Future work will concentrate on the development of systems that identify compound boundaries based on the compound identification systems.

Ultimately, we would like to use these systems to identify potential compounds from large plain text corpora. For Dutch we plan to identify compounds in the SoNaR corpus (Oostdijk et al., 2008; Oostdijk et al., In press). SoNaR is a large collection of written contemporary Dutch texts. It contains approximately 500 million tokens and as such it provides a good starting point to identify naturally occurring compounds in Dutch.

Acknowledgments

The research described in this paper has been performed in the context of the Automatic Compound Processing (AuCoPro) project. This is a collaboration between researchers from North-West Univer-

sity (Potchefstroom, South Africa), the University of Antwerp (Belgium) and Tilburg University (The Netherlands). The project concentrates on the identification of compound boundaries (Tilburg) and the semantic relations between the elements in compounds (Antwerp). This research is performed both on Dutch and Afrikaans (Potchefstroom).

The project is co-funded by a joint research grant of the Nederlandse Taalunie (Dutch Language Union) and the Department of Arts and Culture (DAC) of South Africa and a grant of the National Research Foundation (NRF) (grant number 81794).

We would also like to thank Sylvie Bruys, who helped during the initial phase of the project and who has manually annotated compound data.

References

Alfonseca, E., Bilac, S., & Pharies, S. (2008). German decompounding in a difficult corpus. *CICLing Conference on Computational Linguistics and Intelligent Text Processing* (pp. 128–139). Berlin, Heidelberg: Springer.

Booij, G. (1996). Verbindingsklanken in samenstellingen en de nieuwe spellingregeling. *Nederlandse Taalkunde*, *2*, 126–134.

Creutz, M., & Lagus, K. (2005). *Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0* (Technical Report). Helsinki University of Technology.

CText (2005). *Ckarma: C5 kompositumanaliseerder vir robuuste morfologiese analise* (Technical Report). Centre for Text Technology (CText), North-West University, Potchefstroom, South Africa.

Garera, N., & Yarowsky, D. (2008). Translating compounds by learning component gloss translation models via multiple languages. *Proceedings of the 3rd International Conference on Natural Language Processing (IJCNLP)* (pp. 403–410).

Ha, L. Q., Sicilia-garcia, E. I., Ming, J., & Smith, F. J. (2003). Extension of Zipf's law to word and character n-grams for English and Chinese. *Journal of Computational Linguistics and Chinese Language Processing*, *8*, 77–102.

Koehn, P., & Knight, K. (2003). Empirical methods for compound splitting. *Proceedings of the 11th Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL); Dublin, Ireland* (pp. 187–194).

Macherey, K., Dai, A. M., Talbot, D., Popat, A. C., & Och, F. (2011). Language-independent compound splitting with morphological operations. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics; Portland, OR, USA* (pp. 1395–1404). New Brunswick:NJ, USA: Association for Computational Linguistics.

Mima, H., & Ananiadou, S. (2000). An application and evaluation of the c/nc-value approach for the automatic term recognition of multi-word units in japanese. *Terminology*, *6*, 175–194. Special issue on Japanese Term Extraction.

Oostdijk, N., Reynaert, M., Hoste, V., & Schuurman, I. (In press). The construction of a 500-million-word reference corpus of contemporary written dutch. In P. Spyns and J. Odijk (Eds.), *Essential speech and language technology for dutch: Results by the stevin-programme*, chapter 13. Berlin Heidelberg, Germany: Springer-Verlag.

Oostdijk, N., Reynaert, M., Monachesi, P., van Noord, G., Ordelman, R., Schuurman, I., & Vandeghinste, V. (2008). From D-Coi to SoNaR: A reference corpus for dutch. *Proceedings on the sixth international conference on language resources and evaluation (LREC 2008); Marrakech, Marokko* (pp. 1437–1444). ELRA.

Pilon, S., & Puttkammer, M. J. andVan Huyssteen, G. B. (2008). Die ontwikkeling van 'n woordafbreker en kompositumanaliseerder vir afrikaans (the development of a hyphenator and compound analyser for afrikaans). *Literator*, *29*, 21–41.

Ryder, M. E. (1994). *Ordered chaos: The interpretation of english noun-noun compounds*, vol. 123. University of California press.

van Huyssteen, G. B., & van Zaanen, M. M. (2004). Learning compound boundaries for afrikaans spelling checking. *Pre-Proceedings of the Workshop on International Proofing Tools and Language Technologies; Patras, Greece* (pp. 101–108).

Wiese, R. (1996). *The phonology of german*. New York: NY, USA: Oxford University Press.

Ziou, D., & Tabbone, S. (1998). Edge detection techniques - an overview. *International Journal of Pattern Recognition and Image Analysis*, *8*, 537–559.

Compression-based inference on graph data

Peter Bloem P@PETERBLOEM.NL
System and Network Engineering Group, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands

Keywords: compression, machine learning, graphs, Kolmogorov complexity, minimum description length, normalized compression distance, subdue

Abstract

We investigate the use of compression-based learning on graph data. General purpose compressors operate on bitstrings or other sequential representations. A single graph can be represented sequentially in many ways, which may influence the performance of sequential compressors. Using Normalized Compression Distance (NCD), we test a sequential compressor versus a native graph compressor. We use both synthetic, randomly generated graphs and real-life datasets. We conclude that, even under adverse circumstances, sequential representations contain enough structure for shallow algorithms to perform inference successfully. Algorithms that operate directly on the graph representation usually require a considerable increase in resources, but do allow for an increase in performance also.

The mantra of compression-based inference is that *compression equals learning*. If we can compress data, we must have found some structure, we must have learned something. Conversely, if we have learned something about data, we should be able to use that knowledge to represent our data more succinctly. [1]

The idea of compression-based learning is expressed in two related subjects: *Minimum Description Length* and *Kolmogorov Complexity*. Minimum Description Length (Grünwald, 2007) builds a statistical framework on the principle that a good model is one that can be used to describe the data in as few bits as possible. Kolmogorov Complexity (Li & Vitanyi, 1997) is concerned with the mathematical expression of the *information content* of data. It states that if we can find some short description of a dataset (ie. compress it) then the total information content must be below the length of that description.

In this paper we investigate the use of these compression-based methods on graph data. Examples of such data are social graphs, transportation graphs, trade networks or semantic graphs. The graph is a powerful and versatile representation. Most applications of compression-based learning use sequential models, such as deterministic finite automata or block-sorting compressors, which operate on bitstrings. If we have data in the form of a graph, we can translate it to a bitstring, of course, but in this transformation we complicate our problem. For different orderings of nodes, the graph is the same, but the bitstring changes radically. To a simple compressor, the two bitstrings may not share very much structure, even though they represent the same graph. We will call this the problem of isomorphism.

If this issue really 'blinds' the sequential compressors to the structures in the graph, one option is to investigate compressors that operate on the level of the graph representation, for instance by finding frequent subgraphs or clustering the graph. The methods do not suffer this problem of isomorphism, but as a result they are more expensive than their sequential cousins. In this paper we hope to provide a first indication of how far the sequential approach will go, and whether the native approach will let us continue on from there.

There is a wealth of research available on machine learning and data mining both within single graphs

[1] Other features commonly associated with learning, such as generalization will, so the adherents of compression-based learning argue, be optimized along with the optimization of compression. See, for instance Grünwald, 2005; Grünwald, 2007.

Appearing in *Proceedings of BENELEARN 2013*.

and on sets of graphs (see for instance Cook & Holder, 2006). In this paper we do not share the data mining goal of extracting interesting features and models from the data, but only the goal of performing inference, using the compressors and their learning techniques as black boxes, and evaluating the the results of a chosen inference task.

Normalized Compression Distance

From the many machine learning methods based on principles of compression, we choose *Normalized Compression Distance* (NCD) as a representative method of compression-based learning. The choice is a practical one: NCD is a simple method which requires nothing more than a general purpose compressor. Any domain-specific knowledge we wish to use about our data (eg. it represents a graph) can be added to the compressor. We proceed under the assumption that conclusions reached about the performance of NCD on graph data will translate to MDL and other frameworks.

We will provide a brief, intuitive explanation of the principles involved, sufficient to understand the ideas presented in this paper. For a more in-depth and rigorous treatment we refer the reader to (Li et al., 2004) and (Cilibrasi & Vitányi, 2005). For a general introduction to Kolmogorov complexity, see (Li & Vitanyi, 1997).

Kolmogorov complexity

Kolmogorov complexity is a notion of information content based on two principles: (a) all data can be represented as a bitstring (b) the shorter this string can be described, the less information is contained in it.

The second principle is formalized in two steps. First, by 'description' we mean a description in a formal language that is maximally expressive. To formalize expressiveness, we require that the method of description is Turing-complete (of equivalent strength to a Turing machine). By the strong Church-Turing thesis, this suggests that there is no reasonable way of defining a more expressive method of description.

We choose the Turing machine as a canonical method of description, and we fix some enumeration of all Turing machines $\{T_i\}$. There exists a Universal Turing machine U that is defined as follows:

$$U(\langle i, p\rangle) \simeq T_i(p)$$

That is, if U's input consists of two bitstring arguments i and p, combined with some computable pairing function $\langle\cdot,\cdot\rangle$, then U computes the same function as T_i on input p if $T_i(p)$ halts, or fails to halt if T_i fails to halt. U provides our true formalization of 'description'. If our data is x, and for some $y = \langle i, p\rangle$, $U(y) = x$, we say that y is a description of x on our reference universal Turing machine.

We can now say that with respect to U, there must be a minimal description for any given data:

$$K_U(x) = \min\{|y| : U(y) = x\}$$

The result that makes Kolmogorov complexity a useful measure of information content is that $K_U(\cdot)$ is only marginally dependent on the choice of U. If we suppose that there is another universal description method, V, we might ask what the expected difference is between $K_U(x)$ and $K_V(x)$. Let $k_V(x)$ is the shortest program for x on V. Since, U is universal, we know that it can compute k_V by simulating V. Somewhere in our enumeration of Turing machines $\{T_i\}$, there is a T_ν which computes the same function as V. This simulation is a program for x on U which is a bound for the shortest program on U:

$$\begin{aligned} K_U(x) &\leqslant |\langle T_\nu, k_V(x)\rangle| \\ &= |k_V(x)| + |\nu| + p(\nu) \\ &= K_V(x) + O(1) \end{aligned}$$

Where $p(\nu)$ is the penalty (ie. the additional bits) that the pairing function requires to store its two arguments in a separable way. We require that this is only dependent on ν. The final line shows that $K_U(\cdot)$ and $K_V(\cdot)$ differ only by a constant term, which is independent of x. To summarize: we may differ in opinion on how much information our data contains, but only by a constant amount.

In some contexts, it is desirable to distinguish between the classical Kolmogorov complexity and the prefix-free Kolmogorov complexity. In the context of Normalized Compression Distance the distinction does not matter.

A complete treatment of Kolmogorov Complexity is outside the scope of this paper, but the following properties are important to understand.

$K(\cdot)$ is uncomputable There can be no algorithm which computes the Kolmogorov complexity of x for all x. It can, however, be bounded from above, and for every algorithm which bounds it, there is another algorithm which provides a better bound.

All computable compressors approximate $K(\cdot)$ If we have some compressor for our data x (say

GZIP) we can find the decompression algorithm somewhere in $\{T_i\}$, say as T_g. We can have a description on U as $U(\langle g, z\rangle) = x$ so that $K_U(x) \leqslant |z| + O(1)$. In this way, any computational structure in x is taken into account in $K(x)$, and $K(\cdot)$ can be approximated by any computable compressor.

This gives us the basic philosophy behind all translations of Kolmogorov complexity to the realm of practical applications: we approximate Kolmogorov complexity by some learning algorithm or compressor.

Finally, we define conditional Kolmogorov Complexity $K(x \mid y)$. Where regular Kolmogorov Complexity is defined as the shortest program which produces x, the conditional variant is defined as the shortest program which produces x when provided with y. A complete treatment is available in (Li & Vitanyi, 1997).

Normalized Information Distance (NID)

The length of the shortest program to get from y to x intuitively suggests that $K(\cdot \mid \cdot)$ can be seen as a similarity measure. Clearly, very little is required to transform a string into itself, or into a very similar string, whereas for two random strings, only a program that stores the second in its entirety can make the transformation.

This intuition prompted Li and Vitányi (Li et al., 2004) to investigate the use of Kolmogorov Complexity as a metric of computational similarity. To acquire a true metric, some problems have to be solved. The first is that $K(\cdot \mid \cdot)$ is not symmetric: it takes a small program to blank out the collected works of Shakespeare, but the reverse is a more complex operation. The first step, then, is to define the (symmetric) Information Distance:

$$ID(x, y) = \max\left[K(x \mid y), K(y \mid x)\right]$$

The second issue is one of scale. If two strings of a million bits differ by 1000 bits, we might consider them quite similar, whereas two strings of 1000 bits that differ by that amount could not be more different.[2] Therefore, we would like to take the length of the strings into account. This gives us the Normalized information Distance (NID)

$$NID(x, y) = \frac{\max\left[K(x \mid y), K(y \mid x)\right]}{\max\left[K(x), K(y)\right]}$$

[2]Note that this is only an intuitive example. If two strings differ in exactly every bit, a very short program transforms one into the other, so by NID, they are very similar.

We would like to approximate this by replacing each occurrence of the Kolmogorov complexity with an approximation by a compressor, which we will call C. As most compressors do not work on a conditional basis (expressing data given some existing data), we want to rewrite the conditional K's as nonconditional ones. To achieve this, we accept beyond the constant term uncertainty that is innate to Kolmogorov Complexity, a further logarithmic inaccuracy. This allows us to rewrite as

$$\begin{aligned} NID(x, y) &= \frac{\max\left[K(x, y) - K(x), K(y, x) - K(y)\right]}{\max\left[K(x), K(y)\right]} \\ &= \frac{\max\left[K(xy) - K(x), K(yx) - K(y)\right]}{\max\left[K(x), K(y)\right]} \end{aligned}$$

If we replace the Kolmogorov complexity with a compressor C, we get the normalized compression distance

$$NCD(x, y) = \frac{C(xy) - \min\left[C(x), C(y)\right]}{\max\left[C(x), C(y)\right]}$$

This step also includes the assumption that our compressor is roughly symmetric ($C(xy) = C(yx)$).

When our data is represented as a graph, rather than a string, we replace the notion of concatenation of strings by concatenation of graphs. That is, we combine the graphs x and y into a single (disconnected) graph.

Methods

Our aim is to test a sequential and a graph-based compressor on an inference task for a variety of graph data. To ascertain the performance of the compressors, we generate graphs from different sources, calculate their NCD distances and see whether a clustering algorithm can reconstruct the original sources as clusters. Datasets and source code for these experiments are available. [3]

Node ordering

An important and subtle concern is the ordering of nodes in the sequential representation of our graphs. This issue is detailed very well by Kang & Faloutsos, 2011. As shown, there are various algorithms to determine node orderings that bring out a lot of the graph's inherent structure in the adjacency matrix, allowing a sequential compressor to exploit it. We could use substantial resources to find a good ordering of nodes to

[3]`http://www.peterbloem.nl/benelearn2013`

improve the performance of the sequential compressor. If we did, however, the extra computation might mean that the compressor is no longer a shallow model. To maintain the sequential compressor as a representative example of shallow models, the node ordering should be cheap to establish from a random ordering, preferably in linear time.

Since we are testing the capacity of the general purpose compressor to perform inference despite the problem of isomorphism, we will actually present it with a worst case scenario. We use a random ordering of nodes for all graphs. If the general purpose compressor still outperforms the random baseline under these conditions, it will tell us that it is, at least in part, resistant to the problem of isomorphism.

Experiment 1: Synthetic data

We generate graphs from four models.

The first is the classic Erdős-Rényi (ER) model, where a uniform random choice is made from all graphs with n nodes and m links. The second is the Barabási-Albert (BA) model (Albert & Barabási, 2002), which grows a graph from a set of n_0 unconnected nodes, one node at a time, connecting each new node to k distinct existing nodes where the probability that a given existing node is chosen for a connection is its degree, divided by the sum of the degrees of all nodes. Thus, under the BA model the more links a node has, the higher the probability that it will accrue even more. This effect causes the degree distribution of a BA network to become scale-free (ie. it follows a power law).

Since we want there to be some challenge in separating the two classes of network, we ensure that they have the same number of nodes and links. To achieve this, we first generate the BA networks, count their nodes and links and use these as parameters for the ER model.

We also include graphs from the fractal graph generation algorithm from (Song et al., 2006). We set the hub-parameter which determines the level of fractality (as a trade-off with the level of small-worldness) to 0.0 (for a small world network) and to 1.0 (for a fractal network).

Once we have this dataset, consisting of four gold clusters, we calculate the NCD with a given compressor for every pair of graphs in the dataset, giving us a symmetric matrix. We use the k-medoids algorithm to cluster this set into four clusters.

To assess the performance of the clustering we label the clusters so that the accuracy is maximized (essentially assigning the optimal labels). Clearly, this would be cheating when testing a classifier, but since we are only interested in the clustering aspect, it gives us a straightforward performance measure.

As a random baseline, we generate a random distance matrix with every distance a uniform random value in $[0, 1)$, and run the clustering algorithm on that.

Experiment 2: Real-life data

In this experiment we sample subgraphs from large, existing graphs. We sample by choosing a random node uniformly from all nodes and performing a random walk of length n. We then extract a subgraph containing the nodes encountered and any links connecting two encountered nodes. We replace all node and link labels with a single canonical symbol.

With this dataset of subgraphs, we proceed as before, calculating the distances between the subgraphs and clustering them into as many clusters as we have sources, to see whether the resulting clusters match the sources.

We use the following datasets:

cellular The cellular network of the E. Coli bacterium. (Jeong et al., 2000) Acquired from `http://www.nd.edu/~networks/resources/metabolic/index.html`

neural The neural network of the C. Elegans nematode worm (ignoring link directions). (Achacoso & Yamamoto, 1991; Watts & Strogatz, 1998) Acquired from `http://toreopsahl.com/datasets/#celegans`

co-purchase A graph of items commonly purchased together on internet retailer Amazon.com. (Leskovec et al., 2007) Acquired from `http://snap.stanford.edu/data/amazon0302.html`

Compressors

GZIP

We use GZIP as our general purpose compressor. Specifically, in our experiments, we use the implementation of GZIP that is part of the standard Java SDK. To store a graph with GZIP, we flatten the lower half of its adjacency matrix into a bitstring and store this together with a list of the node and link labels. We use Java object serialization to take care of delimiting the label data and translating it to bits. (Since all graphs in our experiments have a single label, this is unlikely to affect the outcome).

Subdue

Subdue (Jonyer et al., 2004; Ketkar et al., 2005) is an algorithm for finding frequent subgraphs in graph data. The algorithm searches for the subgraph that maximally compresses the data. The body of the algorithm is essentially a beam search through the space of subgraphs. It consists of three main routines:

Subgraph matching This is an algorithm for finding the occurrences of a given subgraph in a graph. The method used is detailed by Bunke & Allermann, 1983. Since this is a semi-exhaustive search for the solution to the NP-complete problem of subgraph isomorphism, the matching can only be solved for very small subgraphs. Unfortunately, even with subgraphs of four or five nodes, the matching is too slow in combination with the number of times it is executed to calculate a full distance matrix. To combat this issue, we remove all but the first $\mathsf{b}_{\text{inner}}$ elements from the search queue after it is sorted at each iteration, effectively turning the algorithm into a beam search.

MDL Scoring This routine takes a subgraph, finds its occurrences in the data by the previous routine and deletes these from the data. The subgraph is then stored once, together with the remainder of the data and a list of where the subgraph should be attached to reconstruct the original data. See the appendix for an exact description.

Subgraph search This is the 'outer loop' of the algorithm. Starting with a graph of a single link between two nodes, it searches through the space of all connected graphs by extending each current candidate by one link at a time (possibly by adding a new node as well). The buffer of current candidates is sorted by MDL score, and the candidate with the highest score is extended to create new candidates. At each iteration all but the top $\mathsf{b}_{\text{outer}}$ candidates are removed, turning the search into a beam search.

Algorithm 1 shows a broad description of the whole procedure.

The graph matching search (the first line of the **score** function) allows for inexact matches of the subgraph. In these cases, we use a rough upper bound of the number of bits required to transform the stored subgraph into the subgraph that is actually present in the data.

Algorithm 1 Pseudocode for the Subdue algorithm

```
G: the data graph
b_outer: the beam size

S ← [K_1] # initialize the list of substructures with a graph of a single node

loop
    s ← head element of S
    add all extensions of s to S
    sort S by score(s', G) for s' ∈ S
    remove all but the first b elements of S

function score(s', G)
    replace occurrences of s' in G with node N
    annotate links to N with the nodes in s'
    return nr of bits to store the edited G and s'
```

Parameter settings and specifics

All graphs generated contain 100 nodes. In the BA-model, we attach one node each step, giving 100 links also. Note that this makes the BA graphs UAGs. For more nodes attached per step the clustering problem would become more difficult. The random graphs are generated with the exact same number of nodes and links. The fractal graphs we generate to depth 2 by adding 4 ancestors at each side of each link and 1 extra link between the groups of ancestors. This results in networks of 90 nodes and 100 links.

For each source, we generate 3 graphs.

The Subdue algorithm has a lot of parameters. During the search we return only one best subgraph. Our beam width at the top level ($\mathsf{b}_{\text{outer}}$) is set to 5. The beam width in the graph matching routine ($\mathsf{b}_{\text{inner}}$) is set to 10. We run the search for 10 iterations, limiting the size of the subgraph used to 5.

We let the k-medoids algorithm run for 20 iterations. This is more than enough for convergence in all experiments.

Results

Table 1 shows the results on randomly generated graphs. Table 2 shows the results for subgraphs sampled from real-life datasets.

Conclusions and future work

Our experiments show that sequential, general purpose compressors are better at performing graph inference than expected. Despite the random ordering of the nodes, the bitstring contains enough shallow patterns that a compressor like GZIP can tell the two types of fractal graphs apart, and only struggles with the dif-

Table 1: Confusion matrices for various compressors. Columns represent the clusters found by the k-medoids algorithm. To calculate the error, we label the resulting clusters so that the error is minimized (ie. reorder the columns of the confusion matrix to maximize the sum of the diagonal). We report the mean error (1 - the sum of the diagonal) over 10 experiments (and the standard deviation in brackets) below each confusion matrix. The confusion matrix shown is always for the first experiment in the run.

ER	0.17	0.083	0	0
BA	0.083	0.17	0	0
fractal (pure)	0.083	0	0.17	0
fractal (small world)	0	0	0	0.25

(a) Random baseline: error 0.46 (0.11)

ER	0	0.25	0	0
BA	0	0.25	0	0
fractal (pure)	0	0	0.25	0
fractal (small world)	0	0	0	0.25

(b) GZIP: error 0.27 (0.12)

ER	0.25	0	0	0
BA	0	0.25	0	0
fractal (pure)	0	0	0.25	0
fractal (small world)	0	0	0	025

(c) Subdue: error 0.14 (0.14)

Table 2: Results for the experiment on natural datasets.

cellular	0.11	0.11	0.11
neural	0.11	0.11	0.11
co-purchase	0.11	0	0.22

(a) Random baseline: error 0.43 (0.11)

cellular	0.33	0	0
neural	0.22	0.11	0
co-purchase	0	0.22	0.11

(b) GZIP: error 0.28 (0.17)

cellular	0.33	0	0
neural	0	0.33	0
co-purchase	0.33	0	0

(c) Subdue: error 0.34 (0.17)

ference between the random and BA graphs. This is particularly interesting considering the high resource requirements of most algorithms for graph inference, and the low resource use of general purpose compressors.

This result suggests that at least some inference on graphs can be performed by sequential algorithms on a sequential representation in linear time, with decent results.

As for the graph-compressors, we see a small improvement relative to the sequential compressors for a strong increase in computational resources. Subdue as used in this paper is a relatively simple compressor, which isolates only a single subgraph for compression and we tested it only at modest parameters. The publications surrounding Subdue offer much more complex solutions (such as the induction of graph grammars (Jonyer et al., 2004)). To investigate the promise of these models as compressors further, it will be necessary to investigate both parallelized versions of these algorithms and a more elegant relaxation of the exhaustive nature of their components. Subgraph sampling methods like the ones detailed by Kashtan et al., 2004, may be able to provide a significant increase in performance.

The notion of compression-based learning is a good framework within which to combine many approaches to inference from the most general to the most domain specific. The Minimum Description Length principle and its associated techniques, which have not been investigated yet for reasons of scope, offer the promise of an even broader field of approaches to the analysis of graph data.

Acknowledgements

This publication was supported by the Dutch national program COMMIT. The author would like to thank Leen Torenvliet, Eugenio Bargiacchi and Eugenio di Leo for comments and preliminary research.

We would like to thank the reviewers for many insightful comments.

Appendix: Graph coding

The method of coding data is always a sensitive point in compression-based learning. The precise choices made in translating the data to a string of bits can greatly affect which patterns are picked up, or ignored by the subsequent inference procedure. Here we detail the procedure used to encode the graphs in both compressors.

GZIP

We take half of the adjacency matrix ($(\mathtt{n}^2+\mathtt{n})/2$ bits), and store it in flattened form as an array of java byte primitives, together with the size of the string (since the stored representation is padded to a multiple of eight). We then serialize this representation into a java GZIPOutputStream. To make our implementation generic, we assume that the nodes and links are labeled, and serialize the labels in a fixed order after the adjacency matrix. In the graphs mentioned in this paper, there is a single label assigned to all elements, so inference shouldn't be affected by the labels.

Subdue

In the subdue case we do our coding in a more precise way, without relying on platform-specific functions. More importantly, we only count the bits required to store our graph, rather than actually constructing the representation itself.

Plain graph

To store a plain graph, we follow roughly the coding strategy outlined in Holder et al., 1994.

We first store the number of nodes $\mathtt{n}$ in prefix free coding, and the maximum number of neighbours for a node in the graph $\mathtt{n}_{\max}$ (in $\log \mathtt{n}$ bits). We then store the lower half of the adjacency matrix (including the diagonal) row by row.

For node $\mathtt{i}$, we only need to store the connections to the $\mathtt{i}$ nodes below it including itself. We use $\log \mathtt{n}_{\max}$ bits to store the number of such neighbours $\mathtt{n}_\mathtt{i}$, and $\log \binom{\mathtt{i}}{\mathtt{n}_\mathtt{i}}$ bits to store the configuration of those neighbours.

After this, all that remains is to encode the labels of the nodes and links. We assume that the sender and receiver in our coding scheme possess a codebook that is efficient for the data given, so that if a label $\mathtt{l}$ occurs with frequency $\#\mathtt{l}$, we can encode it in $-\log \frac{\#\mathtt{l}}{\sum_\mathtt{k} \#\mathtt{k}}$ bits.

We assume that there is some canonical ordering among the nodes and links, and store them as a stream. Since the number of labels is known from the adjacency matrix, the code for the entire graph is self-delimiting.

Graph with substructure

To store the graph with a substructure, we first store the substructure itself. Since this contains no special symbols, we can store it simply using the above method (except we use the codebook based on the whole graph rather than the substructure to encode the labels). This is a prefix code, so we can start encoding the rest of the graph right after.

In the rest of the graph, we remove the nodes matched to the substructure and all links connecting to them. We store the 'silhouette' of the substructure as a plain graph (again with the codebooks of the whole graph).

We then store the way the substructures should be connected into the silhouette to reconstruct the original graph. For each substructure we first store the transformation cost (if the substructure was an inexact match), with a prefix penalty to make the description self-delimiting, we then store the number of links connecting the substructure un prefix-coded form, and then for each link we take $\log \mathtt{s}$ bits to encode how to connect it in the substructure and $\log \mathtt{d}$ to connect it in the rest of the graph, where $\mathtt{s}$ is the number of nodes in the substructure and $\mathtt{d}$ is the number of nodes in the silhouette.

In later versions of our code, the occurrences of the subgraph are replaced by symbol nodes, but the version used to perform these experiments uses the silhouette method.

References

Achacoso, T. B., & Yamamoto, W. S. (1991). *Ay's neuroanatomy of c. elegans for computation*. CRC.

Albert, R., & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of modern physics*, *74*, 47.

Bunke, H., & Allermann, G. (1983). Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, *1*, 245–253.

Cilibrasi, R., & Vitányi, P. M. (2005). Clustering by compression. *Information Theory, IEEE Transactions on*, *51*, 1523–1545.

Cook, D. J., & Holder, L. B. (2006). *Mining graph data*. Wiley-Interscience.

Grünwald, P. (2005). A tutorial introduction to the minimum description length principle.

Grünwald, P. D. (2007). *The minimum description length principle*. MIT press.

Holder, L. B., Cook, D. J., & Djoko, S. (1994). Substructure discovery in the subdue system. *Proceedings of the Workshop on Knowledge Discovery in Databases* (pp. 169–180).

Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., & Barabási, A.-L. (2000). The large-scale organization of metabolic networks. *Nature*, *407*, 651–654.

Jonyer, I., Holder, L. B., & Cook, D. J. (2004). Mdl-based context-free graph grammar induction and applications. *International Journal on Artificial Intelligence Tools*, *13*, 65–79.

Kang, U., & Faloutsos, C. (2011). Beyond'caveman communities': Hubs and spokes for graph compression and mining. *Data Mining (ICDM), 2011 IEEE 11th International Conference on* (pp. 300–309).

Kashtan, N., Itzkovitz, S., Milo, R., & Alon, U. (2004). Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, *20*, 1746–1758.

Ketkar, N. S., Holder, L. B., & Cook, D. J. (2005). Subdue: compression-based frequent pattern discovery in graph data. *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations* (pp. 71–76).

Leskovec, J., Adamic, L. A., & Huberman, B. A. (2007). The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, *1*, 5.

Li, M., Chen, X., Li, X., Ma, B., & Vitányi, P. M. (2004). The similarity metric. *Information Theory, IEEE Transactions on*, *50*, 3250–3264.

Li, M., & Vitanyi, P. M. (1997). *An introduction to kolmogorov complexity and its applications*. Springer Verlag.

Song, C., Havlin, S., & Makse, H. A. (2006). Origins of fractality in the growth of complex networks. *Nature Physics*, *2*, 275–281.

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of small-world networks. *nature*, *393*, 440–442.

Unsupervised Learning of Features for Bayesian Decoding in Functional Magnetic Resonance Imaging

Umut Güçlü U.GUCLU@DONDERS.RU.NL
Marcel van Gerven M.VANGERVEN@DONDERS.RU.NL
Radboud University Nijmegen, Donders Institute for Brain, Cognition and Behaviour, Montessorilaan 3, 6525 HR Nijmegen, The Netherlands

Keywords: Bayesian inference, functional magnetic resonance imaging, independent component analysis, neural decoding, unsupervised feature learning

Abstract

Neural decoding is concerned with inferring certain aspects of a stimulus from brain activity. With the recent advent of functional magnetic resonance imaging (fMRI), it has become possible to create a literal picture of a visual stimulus from the human brain. Most conventional decoders are based either on the input space or on a hand-designed feature space. An alternative to hand-designing a feature space is unsupervised feature learning, which has seen much success in computer vision. Here, we present a new decoder, which combines Bayesian inversion of voxel-based encoding models with unsupervised feature learning (independent component analysis). We validated our decoder by reconstructing images of handwritten digits from human brain activity measured using fMRI, with state-of-the-art accuracy. Our results show that the feature space has a substantial effect on the accuracy of the reconstructions, and independent component analysis provides an effective means to learn feature spaces for neural decoding in fMRI.

1. Introduction

Neural decoding is concerned with inferring certain aspects of a stimulus from stimulus-evoked brain activity. Functional magnetic resonance imaging (fMRI) measures the activity of many separate voxels (i.e. volumetric pixels) in the brain by detecting the associated changes in the blood-oxygen-level-dependent (BOLD) haemodynamic responses. The spatial resolution afforded by fMRI has made it possible to take advantage of the information contained in distributed patterns of activity evoked by a stimulus in order to classify (Haxby et al., 2001; Kamitani & Tong, 2005; van Gerven et al., 2010a), identify (Mitchell et al., 2008; Kay et al., 2008) or reconstruct (Thirion et al., 2006; Miyawaki et al., 2008; Naselaris et al., 2009; Nishimoto et al., 2011; van Gerven et al., 2010b; van Gerven & Heskes, 2012) the original stimulus.

In the context of neural decoding, reconstruction refers to creating a literal picture of a stimulus from the brain. That is, given an encoding relationship that characterizes how a stimulus or some stimulus features are represented by brain activity, reconstruction is the process of determining the inverse of the encoding relationship (i.e. the decoding relationship) in order to reproduce the stimulus.

Inverting a neural response function is non-trivial because of the stochastic dynamics of neural processes (Brown et al., 2004). Therefore, the encoding relationship is described by a stochastic model (Dayan & Abbott, 2001). Furthermore, prior information about the stimulus is often incorporated in the process of reconstruction, which can also be described by a stochastic model, in order to capture the statistical properties of the environment (Dayan & Abbott, 2001).

Bayesian decoding combines the encoding relationship (i.e. the likelihood) and the prior information (i.e. the prior) using Bayes' theorem in order to describe the decoding relationship (i.e. the posterior). The conventional approach to Bayesian decoding is to characterize how certain "hand-designed" features of a stimulus (e.g. Gabor features) are represented by brain activity.

Appearing in *Proceedings of BENELEARN 2013*.

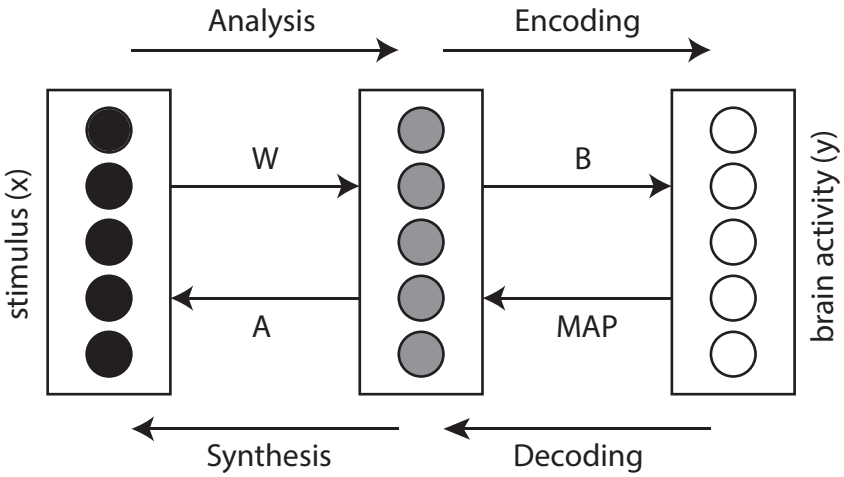


Figure 1. Our framework for reconstructing images from stimulus-evoked BOLD haemodynamic responses. The matrix of linear feature detectors (W) is the parameter of the statistical generative model. W is learned from unlabeled images. The matrix of linear features (A) is the inverse of W. The matrix of regression coefficients (B) is the parameter of the voxel-based encoding models. B is learned from stimulus-response pairs such that the images are analyzed in order to find their latent independent components. The latent independent components are represented by the gray circles in the figure. Reconstruction of a stimulus from stimulus-evoked BOLD haemodynamic responses is the process of maximum a posteriori (MAP) estimation to obtain point estimates of the latent independent components of the stimulus followed by image synthesis to reproduce the stimulus.

However, hand-designing features for complex stimuli and adapting hand-designed features of a particular set of stimuli with certain characteristics to another set of stimuli with different characteristics can be difficult. Unsupervised feature learning is an alternative to the conventional approach, which can mitigate the limitations of hand-designing features and has seen much success in computer vision (Bengio et al., 2012).

Furthermore, while it has been shown that prior information has a substantial effect on reconstruction accuracy (Naselaris et al., 2009), determining a suitable prior that can be used in Bayesian inference has been a challenging goal such that generic priors and empirical priors have often been used (Thirion et al., 2006; Naselaris et al., 2009; Nishimoto et al., 2011; van Gerven & Heskes, 2012). Another advantage of unsupervised feature learning is that a statistical generative model can be used as a prior in Bayesian inference (Hyvärinen et al., 2009). Unsupervised feature learning has already been used in the context of neural decoding. For example, van Gerven et al., (2010b) reconstructed handwritten digits using deep belief networks.

Here, we introduce a new, more straightforward approach to unsupervised feature learning for neural decoding that mitigates the limitations of hand-designing features and gives a proper prior that can be used in Bayesian inference. Our framework combines unsupervised feature learning with Bayesian decoding for reconstructing images from stimulus-evoked BOLD haemodynamic responses (Figure 1).

In particular, we use independent component analysis (ICA) to define a statistical generative model that describes how images are generated as linear transformations of their latent independent components (Hyvärinen, 2010) and linear regression to define voxel-based encoding models that characterize how latent independent components of images are represented by BOLD haemodynamic responses. That is, combining the analysis-synthesis loop and the encoding-decoding loop, reconstruction is defined as the process of Bayesian inference from the voxel-based encoding models followed by image synthesis from the statistical generative model.

In order to learn useful linear features from unlabeled data, to be used in linear regression, we have to impose constraints on the statistical generative model. Two approaches that are typically used is to impose a bottleneck to learn an under-complete representation (van Gerven & Heskes, 2010) and constrain the representation to be sparse (Olshausen & Field, 1996). The statistical generative model defined using ICA discovers interesting structure in the data by learning under-complete non-Gaussian (i.e. sparse) representations.

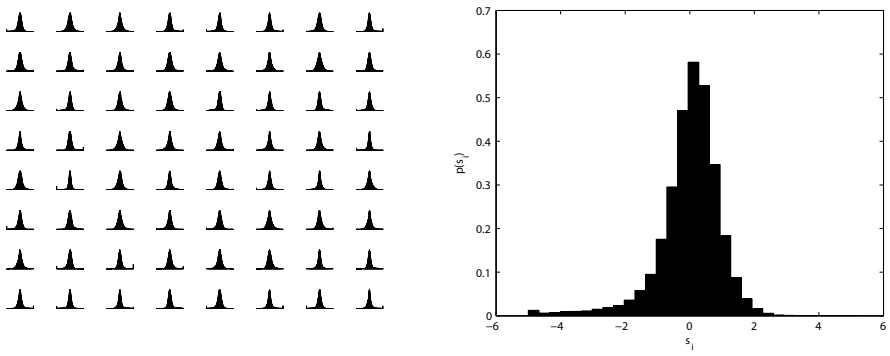


Figure 2. Left panel shows the distributions of 64 latent independent components estimated from grayscale images of handwritten digits. The x-axes represent s_i and the y-axes represent $p(s_i)$. The right panel shows the distribution of one the component in more detail. Note that the distributions are indeed peaked at zero and have high kurtosis.

In the following sections, we first present the derivation of our framework. We then validate our framework by reconstructing grayscale images of handwritten digits from stimulus-evoked BOLD haemodynamic responses. We finally show the effect of unsupervised feature learning on the reconstruction accuracy.

2. Methods

Let $\mathbf{x} \in \mathbb{R}^p$ and $\mathbf{y} \in \mathbb{R}^q$ be a stimulus-response pair where $\mathbf{x}$ is a vector of pixel gray-scale values in an image, and $\mathbf{y}$ is a vector of multiple voxel activities evoked by $\mathbf{x}$. Furthermore, let $\phi : \mathbb{R}^p \to \mathbb{R}^m$ be an invertible linear transformation between the stimulus space and a feature space.

Without loss of generality, we assume that both $\phi(\mathbf{x})$ and $\mathbf{y}$ are normalized to have zero mean and unit variance.

We are interested in the problem of reconstructing $\mathbf{x}$ from $\mathbf{y}$:

$$\hat{\mathbf{x}} = \phi^{-1}\left(\arg\max_{\phi(\mathbf{x})} \{p\left(\phi\left(\mathbf{x}\right)|\mathbf{y}\right)\}\right) \tag{1}$$

where $\hat{\mathbf{x}}$ is a reconstruction of $\mathbf{x}$, and $p(\phi(\mathbf{x})|\mathbf{y})$ is a decoding distribution. We can equivalently formulate the problem of reconstructing $\mathbf{x}$ from $\mathbf{y}$ using Bayes' theorem:

$$\hat{\mathbf{x}} = \phi^{-1}\left(\arg\max_{\phi(\mathbf{x})} \{p\left(\mathbf{y}|\phi\left(\mathbf{x}\right)\right) p\left(\phi\left(\mathbf{x}\right)\right)\}\right) \tag{2}$$

where $p(\mathbf{y}|\phi(\mathbf{x}))$ is an encoding distribution, and $p(\phi(\mathbf{x}))$ is a prior. Therefore, in order to solve the problem of reconstructing $\mathbf{x}$ from $\mathbf{y}$, we need to define ϕ, $p(\phi(\mathbf{x}))$ and $p(\mathbf{y}|\phi(\mathbf{x}))$.

2.1. Unsupervised Feature Learning

We start by defining a statistical generative model of images. Assuming that an image is generated by a linear superposition of some features, we use ICA to define the statistical generative model of images by a linear transformation of the latent independent components of the image:

$$\mathbf{z} = \mathbf{A}\mathbf{s} \tag{3}$$

where $\mathbf{z} \in \mathbb{R}^p$ is a vector of pixel gray-scale values in an image, $\mathbf{A} \in \mathbb{R}^{p \times m}$ is a matrix of linear features, and $\mathbf{s} \in \mathbb{R}^m$ is a vector of the latent independent components of $\mathbf{z}$ such that $m \leq p$. In order to compute s_i as a linear function of $\mathbf{z}$, we invert the linear system defined by $\mathbf{A}$:

$$s_i = \mathbf{W}\mathbf{z} \tag{4}$$

where $\mathbf{W} \in \mathbb{R}^{m \times p}$ is a matrix of linear feature detectors such that $\mathbf{W} = \mathbf{A}^{-1}$. Furthermore, we make the simplifying assumption that s_i have unit variance in order to make s_i unique, up to a multiplicative sign.

We then define $p(s_i)$, assuming that the s_i are non-Gaussian and sparseness is the most dominant type of non-Gaussianity in the images that we are considering (Fig. 2), we use a distribution that is peaked at zero and has high kurtosis to define $p(s_i)$. In particular, we use the logistic distribution:

$$p(s_i) = \text{logistic}\left(0, \frac{\sqrt{3}}{\pi}\right) \tag{5}$$

We then factorize $p(\mathbf{s})$ as the prior on individual s_i:

$$p(\mathbf{s}) = \prod_{i=1}^{m} p(s_i) \tag{6}$$

We can now represent the invertible linear transformation from the input space to the feature space by $\mathbf{W}$ (i.e. $\phi(\mathbf{x}) = \mathbf{s} = \mathbf{W}\mathbf{x}$) and use $p(\mathbf{s})$ as the prior in Bayesian inference (i.e. $p(\phi(\mathbf{x})) = p(\mathbf{s})$).

2.2. Encoding and Decoding

We continue by defining voxel-based encoding models. We use multiple linear regression to define the voxel-based encoding models by a weighted sum of the linear feature detector outputs for responses $1 \leq i \leq q$:

$$y_i = \boldsymbol{\beta}_i^\top \mathbf{s} + \varepsilon_i \tag{7}$$

where $\boldsymbol{\beta}_i \in \mathbb{R}^m$ are vectors of regression coefficients and $\varepsilon_i \in \mathbb{R}$ are Gaussian noise such that $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$. We then use the multivariate Gaussian distribution to define the encoding distribution:

$$p\left(\mathbf{y}|\phi\left(\mathbf{x}\right)\right) = \mathcal{N}\left(\mathbf{B}^\top \mathbf{s}, \boldsymbol{\Sigma}\right) \tag{8}$$

where $\mathbf{B} = (\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_q) \in \mathbb{R}^{m \times q}$ and $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1^2, ..., \sigma_q^2) \in \mathbb{R}^{q \times q}$.

Combining the prior and the encoding distribution using Bayes' theorem results in the decoding distribution:

$$p(\mathbf{s}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{s})p(\mathbf{s}) \tag{9}$$

Having defined the invertible linear transformation from the input space to the feature space and the decoding distribution, we can finally solve the problem of reconstructing $\mathbf{x}$ from $\mathbf{y}$ using maximum a posteriori (MAP) estimation to obtain a point estimate of $\mathbf{s}$ (i.e. $\hat{\mathbf{s}}_{\mathrm{MAP}} = \arg\max_{\mathbf{s}}\{p(\mathbf{y}|\mathbf{s})p(\mathbf{s})\}$) and synthesizing $\hat{\mathbf{x}}$ from the statistical generative model of images (i.e. $\hat{\mathbf{x}} = \mathbf{A}\hat{\mathbf{s}}_{\mathrm{MAP}}$).

2.3. Experimental Validation

For unsupervised feature learning, we used the entire MNIST database of handwritten digits, without the labels (LeCun et al., 1998). That is, the training set consisted of 70000 unlabeled grayscale images of 28 $\times$ 28 pixels in 10 categories (i.e. handwritten zeros through handwritten nines). We preprocessed the images by centering, PCA whitening and dimensionality reduction such that we retained the least possible number of principal components that account for 90% of the variability in the images (i.e. the first 64 principal components). We estimated the parameters of the statistical generative model (Fig. 3) using the FastICA algorithm (Hyvärinen, 1999).

For encoding and decoding, we used the dataset originally published in van Gerven et al. (2010a) and van Gerven et al. (2010b). Briefly, it consisted of estimated peak fMRI responses to grayscale images of handwritten sixes and handwritten nines. The training set consisted of 80 stimulus-response pairs, and the test set consisted of the remaining 20 stimulus-response pairs. Both the training set and the test set had equal number of stimuli from each of the two categories (i.e. handwritten sixes and handwritten nines). We preprocessed the images as in unsupervised feature learning. We trained the voxel-based encoding models using kernel ridge regression and performed hyperparameter optimization using grid search with a nested

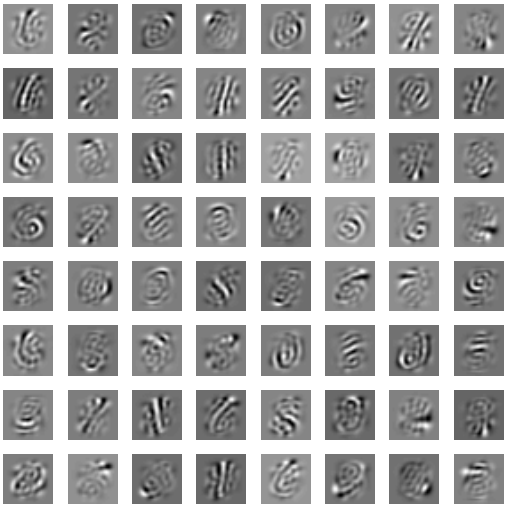


Figure 3. 64 linear feature detectors learned from the MNIST database using the FastICA algorithm.

Figure 4. 64 images synthesized from the statistical generative model by sampling linear feature detector outputs from the estimated distributions of the latent independent components.

Figure 5. Stimuli, ICA reconstructions and PCA reconstructions.

leave-one-out cross validation on the training set. We computed the MAP estimate of the linear feature detector outputs using the minFunc implementation of the limited-memory BFGS algorithm (Schmidt, 2005). For comparative purposes, we used another framework based on preprocessed images (i.e. PCA features), with $\phi(\mathbf{x}) = \mathbf{x}$ and $p(\phi(\mathbf{x})) = \mathcal{N}(0, \mathbf{I}_m)$.

3. Results

We first examined the linear feature detectors (Fig. 3) and synthesized 64 images from the statistical generative model (Fig. 4) by sampling linear feature detector outputs from the estimated distributions of the latent independent components (Figure 2). Visual inspection shows that the linear feature detectors are tuned for meaningful features that resemble pen-strokes, which is consistent with the results in the literature (Ranzato et al., 2006; Lee et al., 2007). That is, the statistical generative model describes images as a linear transformation of pen-strokes. Furthermore, the synthesized images resemble handwritten digits, which suggests that the statistical generative model captures image statistics of handwritten digits.

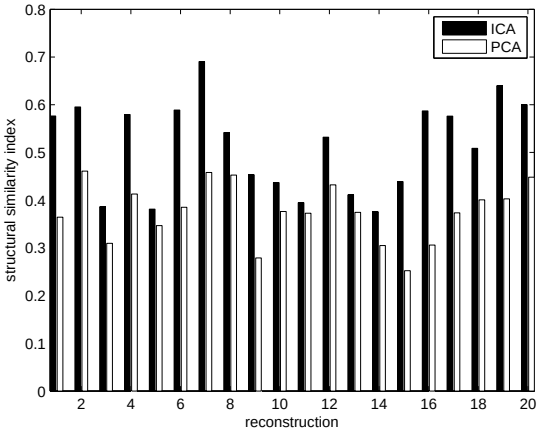


Figure 6. Reconstruction accuracy quantified by computing the structural similarity index.

We then quantified the encoding performance by computing explained variance per voxel. The difference between the mean explained variance of the two frameworks was not significant ($p > 0.05$), with both of the two frameworks having a state-of-the art encoding performance.

We finally evaluated the reconstruction accuracy. Visual inspection shows that our framework has more accurate reconstructions (Fig. 5). We quantified the reconstruction accuracy by computing the structural similarity index (Wang et al., 2004) per reconstruction (Fig. 6). The difference between the mean structural similarity indices of the two frameworks was significant ($p < 0.05$), with our framework having a higher structural similarity index for each of the 20 reconstructions.

4. Conclusion

Here, we introduced a new framework that combines unsupervised feature learning and Bayesian decoding. We validated our framework by accurately reconstructing grayscale images of handwritten sixes and handwritten nines from stimulus-evoked BOLD haemodynamic responses.

The significant improvement in the reconstruction accuracy, but not the encoding performance, demon-

strated the importance of prior information in Bayesian decoding. Using the statistical generative model defined using ICA as a prior in Bayesian decoding results in significantly better reconstructions since ICA captures the image statistics of grayscale handwritten digits better than PCA.

Our framework can be extended beyond grayscale images of handwritten characters, both within the visual modality (e.g. any combination of larger images, natural images, color images, stereo images, temporal sequences of images) and across modalities (e.g. the auditory modality). Furthermore, our statistical generative model can be extended into multiple layers to learn hierarchical features of the stimuli. It remains an open question whether we can accurately reconstruct such complex stimuli from stimulus-evoked BOLD haemodynamic responses.

In conclusion, our results show that the features have a significant effect on the reconstruction accuracy. We also demonstrated that independent component analysis captures the image statistics of grayscale handwritten digits and provides an effective means for unsupervised learning of features for Bayesian decoding in fMRI that can mitigate the limitations of hand-designing features.

Acknowledgments

This work was partially supported by the Academy Assistants Programme of the Royal Netherlands Academy of Arts and Sciences. The authors would like to thank the reviewers for their useful comments.

References

Bengio, Y., Courville, A. C., & Vincent, P. (2012). Unsupervised feature learning and deep learning: a review and new perspectives. *arXiv:1206.5538*.

Brown, E. N., Kass, R. E., & Mitra, P. P. (2004). Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nature Neuroscience*, *7*, 456–61.

Dayan, P., & Abbott, L. F. (2001). *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press.

Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., & Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, *293*, 2425–30.

Hyvärinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, *10*, 626–634.

Hyvärinen, A. (2010). Statistical models of natural images and cortical visual representation. *Topics in Cognitive Science*, *2*, 251–264.

Hyvärinen, A., Hurri, J., & Hoyer, P. O. (2009). *Natural Image Statistics*. Springer London.

Kamitani, Y., & Tong, F. (2005). Decoding the visual and subjective contents of the human brain. *Nature Neuroscience*, *8*, 679–85.

Kay, K. N., Naselaris, T., Prenger, R. J., & Gallant, J. L. (2008). Identifying natural images from human brain activity. *Nature*, *452*, 352–5.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324.

Lee, H., Ekanadham, C., & Ng, A. (2007). Sparse deep belief net model for visual area V2. *Advances in Neural Information Processing Systems*.

Mitchell, T. M., Shinkareva, S. V., Carlson, A., Chang, K.-M., Malave, V. L., Mason, R. A., & Just, M. A. (2008). Predicting human brain activity associated with the meanings of nouns. *Science*, *320*, 1191–5.

Miyawaki, Y., Uchida, H., Yamashita, O., Sato, M.-a., Morito, Y., Tanabe, H. C., Sadato, N., & Kamitani, Y. (2008). Visual image reconstruction from human brain activity using a combination of multiscale local image decoders. *Neuron*, *60*, 915–29.

Naselaris, T., Prenger, R. J., Kay, K. N., Oliver, M., & Gallant, J. L. (2009). Bayesian reconstruction of natural images from human brain activity. *Neuron*, *63*, 902–15.

Nishimoto, S., Vu, A. T., Naselaris, T., Benjamini, Y., Yu, B., & Gallant, J. L. (2011). Reconstructing visual experiences from brain activity evoked by natural movies. *Current Biology*, *21*, 1641–6.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*, 607–9.

Ranzato, M. A., Poultney, C., Chopra, S., & LeCun, Y. (2006). Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems*.

Schmidt, M. (2005). minfunc - unconstrained differentiable multivariate optimization in matlab. www.di.ens.fr/~schmidt/Software/minFunc.html.

Thirion, B., Duchesnay, E., Hubbard, E., Dubois, J., Poline, J.-B., Lebihan, D., & Dehaene, S. (2006). Inverse retinotopy: inferring the visual content of images from brain activation patterns. *NeuroImage*, *33*, 1104–16.

van Gerven, M., Cseke, B., de Lange, F. P., & Heskes, T. (2010a). Efficient Bayesian multivariate fMRI analysis using a sparsifying spatio-temporal prior. *NeuroImage*, *50*, 150–61.

van Gerven, M., de Lange, F. P., & Heskes, T. (2010b). Neural decoding with hierarchical generative models. *Neural Computation*, *22*, 3127–42.

van Gerven, M., & Heskes, T. (2010). Sparse orthonormalized partial least squares. *Benelux Conference on Artificial Intelligence*.

van Gerven, M., & Heskes, T. (2012). A linear Gaussian framework for decoding of perceived images. *2012 Second International Workshop on Pattern Recognition in NeuroImaging* (pp. 1–4).

Wang, Z., Bovik, A. C., Sheikh, H. R., & P Simoncelli, E. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*, 600–612.

Identifying Motifs in Folktales using Topic Models

Folgert Karsdorp FOLGERT.KARSDORP@MEERTENS.KNAW.NL
Meertens Institute, Joan Muyskenweg 25, 1096 CJ Amsterdam, The Netherlands

Antal van den Bosch A.VANDENBOSCH@LET.RU.NL
Radboud University Nijmegen, P.O. Box 9103, 6500 HD Nijmegen, The Netherlands

Keywords: L-LDA, Topic Model, motifs, multi-labeled corpora, ranking, Okapi BM25

Abstract

With the undertake of various folktale digitalization initiatives, the need for computational aids to explore these collections is increasing. In this paper we compare Labeled LDA (L-LDA) to a simple retrieval model on the task of identifying motifs in folktales. We show that both methods are well able to successfully discriminate between relevant and irrelevant motifs. L-LDA represents motifs as distributions over words. In a second experiment we compare the quality of these distributions to those of a simple baseline that ranks words using a TF·IDF weighting scheme. We show that both models produce representations that match relatively well to a manually constructed motif classification system used in folktale research. Finally we show that unlike L-LDA, this simple baseline is capable of representing abstract motifs as generalizations over more specific motifs.

1. Introduction

Without the wondering question "What makes your ears so big?", the story of *Little Red Riding Hood* does not feel complete. Likewise, every telling of *Cinderella* should contain a part about the glass slipper and a cruel stepmother who makes the heroine's life miserable. In folktale research such more or less obligatory passages are called motifs. They "have a power to persist in tradition" (Thompson, 1946) and are part of our collective cultural heritage. Motifs play a key role in the classification of folktales into folktale types. For instance, in the authoritative folktale type catalog *The Types of International Folktales* by Aarne, Thompson and Uther (henceforth: ATU catalog) (Uther, 2004) every tale type is accompanied by a sequence of motifs which are the primary descriptive units of that tale type.

The goal of our work is to automatically identify motifs in folktales. This can be cast as a multi-label classification task in which we attempt to assign a set of motifs to unseen, unlabeled folktales. The set of potential labels that can be assigned to a folktale is large, but certain motifs will be more strongly tied to the particular folktale. We therefore conceptualize our task as a ranking problem.

As discussed in more detail by Karsdorp et al. (2012) and illustrated by Figure 1, the motifs in the Dutch Folktale Database follow a power-law like distribution. Recent research makes a strong case for the use of statistical topic models for multi-label datasets with long-tail label distributions as opposed to discriminative methods (Rubin et al., 2012). In this paper we compare the performance of the supervised topic model Labeled LDA (L-LDA) (Ramage et al., 2009) to a 'simple' retrieval model that uses Okapi BM25 as its ranking function. The first question we would like to answer is: How well do both systems perform on a ranking task where the goal is to allocate the highest ranks to the most relevant motifs?

Topic models such as LDA represent topics as distributions over words. Many studies are devoted to methods that aim to measure the quality and interpretability of these topics, which may not be trivial given the unsupervised nature of LDA. However, we are in a position in which we can use predefined labels, as the motifs used in this study are part of a

Appearing in *Proceedings of BENELEARN 2013*.

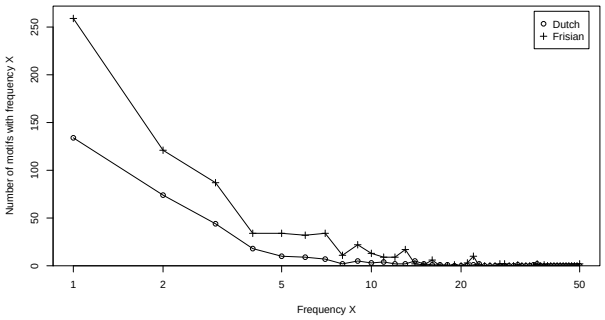


Figure 1. Frequency distribution of motifs on a log scale in Dutch and Frisian folktales in the Dutch Folktale Database.

(hierarchical) classification system, and we have information on which motifs occur in which folktale type. This information is available throughout our data, providing us with ground truth labels. We compare the motif representations discovered by L-LDA to those obtained using a simple baseline in which we compute what words are most strongly associated with each motif using a TF·IDF weighting scheme. We then verify by a quantitative evaluation (using several evaluation metrics from information retrieval) how well the motif representations discovered by both systems compare to a manually constructed motif classification system used in folktale research.

The automatic extraction of motifs is relevant for a number of reasons. Various new folktale digitization initiatives have been undertaken (Meder, 2010; Abello et al., 2012; La Barre & Tilley, 2012), which ask for ways to browse the collections at different facets, such as motifs. This would allow researchers to investigate, for example, how folktales have changed through time in terms of their motif material. It is only since the appearance of Brothers Grimm's version of *Little Red Riding Hood*, for example, that the girl and her grandmother are rescued from the wolf's belly. Extracting motifs from texts also allows researchers to find new relationships between folktales which could tell us more about their evolution.

The outline of the paper is as follows. We will start with providing an overview of related work in Section 2. We then continue with a description of the resources used in this study in Section 3. Sections 4 and 5 are devoted to the experimental setup followed by our results. The last section offers our conclusions and directions for future work.

2. Related work

Voigt et al. (1999) have shown that it is possible to automate motif identification in folklore text corpora by automatically grouping texts based on their content similarities. In their study, the presence of common motifs was derived from co-occurrences of keywords in the texts. For folklore researchers, however, the results are not easily interpretable because motifs are represented as principal components to which no label is assigned.

The literature on multi-label classification is very extensive and has been summarized elsewhere (e.g Tsoumakas & Katakis (2007)). Of special interest for our purposes is the recent work by Nguyen et al. (2013) who showed that Okapi BM25 acts as a competitive baseline in a folktale type classification experiment.

Our work is an application of the multi-label adaptation of Latent Dirichlet Allocation – Labeled LDA – as proposed by Ramage et al. (2009). Rubin et al. (2012) provide an extensive comparison of discriminative multi-label classifiers and three multi-labeled extensions of LDA. They make a strong case for the use of statistical topic models in the context of highly skewed datasets.

Our work differs from both aforementioned papers in three aspects. First, we apply the model to literary texts. It has been observed in many applications that literary texts behave differently from other genres in various ways which requires adaptations of the proposed models. Second, our multi-labeled dataset provides us with the unique possibility to evaluate the topic distributions against ground truth labels. Finally, we will propose a simple way to incorporate the hierarchical structure of our label set into the model.

3. Resources

3.1. TMI and ATU

The comprehensive *Motif-Index* (Thompson, 1955 1958) contains over 45,000 motifs. The motifs are hierarchically ordered in a tree structure. There are 23 alphabetic top-level categories ranging from mythological motifs to motifs concerning traits of character. Many motifs are bound to particular folktale types. Under (1) we list some examples:

(1)Q426 Wolf cut open and filled with stones as punishment;
F911.3 Animal swallows man (not fatally);
F823.2 Glass shoes.

The motifs from the TMI play a key role in the classification of tales into a certain type in the ATU catalog. Every folktale type contains a short summary of the plot. In this summary we find a sequence of motifs that together uniquely identify a folktale. An example of a story summary in the ATU catalog, of the folktale type *The Shepherd Boy*, is as follows.

> ATU 0515, "**The Shepherd Boy.** A foundling child who herds animals finds three objects (of glass) which he gives back to their owners. They promise to reward him [Q42]. With the help of the last owner, a giant, the boy fulfills three tasks. He acquires a castle in which a princess is confined. He rescues her and marries her [L161]."

This tale type contains two motifs, Q42 'Generosity rewarded' and L161 'Lowly hero marries princess'.

3.2. Dutch Folktale Database

The Dutch Folktale Database[1] is a collection of about 42,000 folktales (Meder, 2010). The collection contains folktales from various genres (e.g. fairytales, legends, urban legends, jokes) in a number of variants of Dutch and in Frisian. Every entry in the database contains metadata about the story, including language, collector, place and date of narration, keywords, names, and subgenre. The two largest components contain tales written in standard Dutch and Frisian. In this paper we restrict our experiments to these two components.

Folktales in the Dutch Folktale Database have been manually classified according to the folktale types in the ATU catalog, as far as a link could be established. This link between particular instances of tales and folktale types provides us with the set of motifs that can occur in a folktale type, and therefore in its instantiations. For each folktale in the Dutch Folktale Database that was classified according to the system in the ATU catalog, we assigned to it the set of motifs of its corresponding folktale type.

3.3. Datasets

We created two datasets: one for Dutch folktales and one for the Frisian tales. We only included tales that were classified according to the classification system of the ATU catalog. This resulted in 1,098 Dutch tales and 1,373 Frisian tales. Excluding punctuation, the average number of words per story is 468 for Dutch and 194 for Frisian.

[1] `http://www.verhalenbank.nl`

Both collections were tokenized using the Unicode tokenizer Ucto (Van Gompel et al., 2012).[2] We removed all diacritics and excluded words shorter than two characters and all numbers. As there are no off-the-shelf stemmers available for Frisian, we choose to not do any further preprocessing on the Dutch texts either and use the full tokens.

4. Models

4.1. Baselines

As a baseline for the Dutch and Frisian experiments we use a Big Document Model (see e.g. Nguyen et al. (2013)). For each motif observed in the collection we merge all documents in which that motif occurs into one big document. The ID number of the motif forms the class label of the new document. Given these big documents, we then compute the TF·IDF for all words. We use L2 to normalize the term vectors and smooth the IDF weights by adding one to the document frequencies. This provides us with a ranked list of how strongly a word is associated with a big document, i.e. a motif. We use these ranked lists as a baseline in the cluster evaluation in section 5.2. We will refer to this model as the Big Document Model (BDM).

As a baseline for the ranking experiment in section 5.1 we use a standard retrieval model with Okapi BM25 as our ranking function. BM25 has proven itself to be one of the most successful ranking functions in text-retrieval (Robertson & Zaragoza, 2009). We compute it as follows:

$$S(D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D)+k_1\cdot(1-b+b\cdot\frac{|D|}{\mathrm{avgdl}})} \quad (2)$$

where Q represents a query and $f(q_i, D)$ is the frequency of the i'th term in q in document D. Avgdl is the average document length. The parameters b and k_1 are set to 0.75 and 1.2, respectively. We compute the IDF weight using:

$$IDF(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \quad (3)$$

where N is the number of documents in the corpus and $n(q_i)$ the number of documents that contain q_i. This formulation of IDF can result in negative scores when terms appear in more than fifty percent of the documents. We therefore give the summand in (2) a floor of zero, to filter common terms.

Queries are represented by the complete contents of a test folktale. We issue these queries on the constructed

[2] `http://ilk.uvt.nl/ucto/`

big documents, resulting in a ranking of motifs for that particular folktale.

4.2. Labeled LDA

Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a popular method for extracting topics from texts. LDA is a generative probabilistic model that models documents as distributions over topics. Topics are represented by distributions over words. The model assumes that each word in a document is generated from a single topic.

Ramage et al. (2009) extend the basic framework of LDA by introducing a supervised variant in which the latent topics in LDA correspond directly to the labels assigned to a particular document. Given a corpus of multi-labeled documents the model can estimate the most likely words per label as well as the distribution of labels per document. The primary goal of Ramage et al. (2009) is to show what qualitative advantages L-LDA has over 'traditional' discriminative multi-labeled classifiers such as SVMs. Their results suggest that L-LDA might be advantageous in the context of highly skewed multi-labeled datasets, such as our corpus of folktales (see Rubin et al. (2012) for a more extensive comparison between multi-labeled supervised versions of LDA and SVM classifiers).

In the generative model of L-LDA labels are assumed to be generated from a binomial distribution. As Rubin et al. (2012) point out, in practice L-LDA just assumes the labels to be observed without a prior generative process. For educative purposes they propose a new model – Flat LDA – that does away with this assumption. Our implementation of the model is based on Flat LDA. However, we will still call the model Labeled LDA.

Unlike in unsupervised LDA, we are confident about the labels assigned to a document. To reflect this knowledge, and in order to reduce the variance of the topic distributions, we assign to the labels a relatively high prior ($\alpha = 50$). Because of the relatively small vocabulary size of our corpus, we use a relatively low term smoothing prior ($\beta = 0.001$) to assign the probability mass to only a few words per topic. Both α and β are symmetric priors.

5. Experimental results

5.1. Ranking experiment

In this section we will investigate to what extent we can use L-LDA as a multi-label classifier for the extraction of motifs. We cast the assignment of a set of labels to a document as a ranking task in which the goal is to allocate the highest ranks to the most relevant motifs. We rank the motifs according to their posterior probability in a document. We compare the performance of L-LDA to the retrieval model as described in section 4.1.

Table 1. Evaluation of motif retrieval for BM25 and L-LDA on Dutch and Frisian folktales.

	Model	AP	One Error	Is Error	Margin
Dutch	BM25	0.78	0.26	0.27	10.69
	L-LDA	0.72	0.30	0.39	26.48
Frisian	BM25	0.88	0.15	0.15	4.46
	L-LDA	0.88	0.16	0.16	7.0

We performed 10-fold cross-validation on both datasets, dividing the folktales at random into 10 parts of approximately equal size. As shown by Karsdorp et al. (2012), there are quite many pairs of motifs that co-occur exclusively, that is, they never appear without the other. For these motifs, both models have no way of knowing which words are relevant to which motif as – in information theoretic terms – their mutual information is maximal. We therefore choose to exclude all these informationally indistinguishable motifs from our experiments. Although this results in a rather drastic filtering of motif types, the final number of motif types is still sufficiently high (Frisian: 155, Dutch: 179) and still about eight times higher than in the experiments by Ramage et al. (2009).

In the ideal case, the top of the ranked list contains the motifs of a folktale. The extent to which this is the case reflects how well relevant motifs are found by the systems. We evaluate the ranked lists by means of four evaluation metrics (for reasons of comparability we follow Rubin et al. (2012) in our choice for these evaluation metrics):

Average Precision – Are most or all of the target motifs high up in the ranking?

One Error – For what fraction of documents is the highest-ranked motif incorrect?

Is Error – What fraction of rankings is not perfect?

Margin – What is the absolute difference between the highest ranked irrelevant motif and the lowest ranked relevant motif, averaged across folktales?

The results presented in Table 1 show quite similar results for both L-LDA and BM25. Surprisingly, the

relatively standard retrieval model performs best on all evaluation metrics and on both datasets. In the case of the Frisian folktales the retrieval system is able to emphasize the highest ranks with high precision and a low irrelevance margin. L-LDA produces similar scores but has a slightly higher margin score. Both systems perform better on the Frisian tales than on the Dutch tales. Part of the explanation for this lies in the ratio between motifs and tales in the Dutch collection: there are relatively few folktales with many possible motifs, while the Frisian data has a higher average number of motifs per tale. BM25 shows less sensitivity to this ratio than L-LDA and outperforms L-LDA clearly. In the next section we perform a qualitative analysis to explore why this is the case when we evaluate the motif representations discovered by both models.

5.2. Motif visualization and evaluation

We compare the word distributions discovered by L-LDA to those found by the Big Document Model in which we compute the TF·IDF score for all words in each document. Table 2 shows the top words associated with four motifs for L-LDA and the BDM extracted from Dutch texts (the words are given in their English translation). Many words are discovered by both systems; especially the first few words are found by both methods. However, in some cases L-LDA misses some words characteristic of the given motif. Take motif N211.1.3, 'Lost ring found in fish.' L-LDA ranks the words *fish* and *ring* considerably lower than the BDM.[3]

Standard evaluation of topic identification by LDA is done on the basis of either extrinsic methods (such as retrieval tasks) or intrinsic methods, where the goal is to estimate the probability of test documents or to compute the coherence of topics (see Mimno et al. (2011) and the references cited therein). A rather unique property of the labels under investigation in this study is that they are part of a hierarchical tree structure. A motif such as 'Transformation: pumpkin to carriage' (D451.3.3) belongs to the more abstract category of 'Transformation: object to object' (D450–D499) which in turn is a child motif of the broader parent motif 'Transformation' (D0–D699), which in turn is placed under the top-level node 'Magic' (D), one out of the 23 top nodes.

We perform a hierarchical cluster analysis on the basis of the motifs discovered by L-LDA and evaluate the clusters against the top 23 categories in the hierarchical tree structure of Thompson's *Motif Index*. We choose Ward's method as our linkage method and compute the similarity between motifs using the cosine similarity metric.

[3]It is not necessarily a 'ring' that is found in the fish. There are many variations on this folktale type and often 'teeth' or a 'denture' is found in the fish' belly, which is why BDM ranks these words so high.

Table 3. Clustering results of Dutch and Frisian motif representations.

	Model	homogeneity	completeness	V-measure
Dutch	BDM	0.365	0.330	0.347
	L-LDA	0.344	0.281	0.310
Frisian	BDM	0.354	0.299	0.324
	L-LDA	0.358	0.270	0.308

We evaluate the cluster solution on the basis of three measures (Rosenberg & Hirschberg, 2009):

Homogeneity – Does the cluster solution result in clusters that *only* contain members of the same class?

Completeness – Does the cluster solution result in clusters to which *all* members of the same class have been assigned?

V-measure – An entropy-based measure that expresses the harmonic mean of homogeneity and completeness.

The results in Table 3 show that the quality of the cluster solutions of the two models is quite similar. The solution obtained from the BDM corresponds slightly better to the top-level categorization in the *Motif Index* than the one from L-LDA.

5.3. Exploiting the hierarchical structure of the Motif Index

In the model described above the set of possible motifs was restricted to those motifs that are present in the training data. In the following we describe an extension of the model in which we exploit the relations between motifs in the hierarchical tree of Thompson's *Motif Index*, which lists many motifs not present in the ATU catalog. Yet, because of the hierarchical nature of the index, many ancestral motifs are implicitly observed. The question we would like to explore is: What can we learn about the representation of these more abstract motifs by exploiting the hierarchical structure of the index?

Table 2. The top words within four motifs learned by L-LDA and BDM.

TD·IDF	L-LDA
Q426: Wolf cut open and filled with stones as punishment.	
wolf, Little Red Riding Hood, grandmother, her, children, little kids, your, Oud-Bovetje, big, granny, belly, goat, mother, angry	*wolf, children, mother, door, said, open, her, little kids, so, entire, still, belly, surely, Oud-Bovetje, went*
N211.1: Lost ring found in fish.	
Stavoren, teeth, cod, her, denture, ring, sea, wheat, ships, fish, shipper, harbor, she, the Heerhugowaard	*the, her, and, she, the, of, in, was, a, lady, Stavoren, she, ring, sea, denture*
K343.2.1: The stingy parson and the slaughtered pig.	
clerk, pastor, pig, stolen, will, slaughter, farmers, tonight, everyone, belief, sexton's house, fattened, excellent, slaughter time, pig meat, insignificant	*clerk, pastor, pig, will, said, asked, everyone, stolen, mine, yes, so, must, against*
J2321.1: Parson made to believe that he will bear a calf.	
student, pastor, little bottle, cork, uroscopy, monkey, John, clerk, pregnant, rubber band, quack, butt, give birth, your, spins	*the (de), a, pastor, John, student, the (het), to be, must, water, says, comes, to (te), to (om), and, surely*

Table 4. Clustering results of Dutch and Frisian motif representations (including ancestor motifs).

	Model	homogeneity	completeness	V-measure
Dutch	BDM	0.339	0.315	0.327
	L-LDA	0.159	0.177	0.168
Frisian	BDM	0.414	0.377	0.394
	L-LDA	0.197	0.199	0.198

Figure 2 shows the tale type ATU 333 *Little Red Riding Hood* as a layered sequence of motifs. The gray nodes are observed in the ATU catalog under index ATU 333. The observed motifs inherit certain information from its ancestors. Although we have no direct information about the unshaded motifs in the graph, it should be possible to infer some information about their features. The motifs F911.3 and F913, for example, share the concept of "extraordinary swallowing" and have some idiosyncratic aspects themselves. If we assume that a motif such as F911.3 is a mixture of features from its parents and of its own, we might be able to learn about the features of the unobserved more abstract motifs.

Each folktale is labeled with the motifs that are listed by its corresponding tale type in the ATU catalog. We expand this motif set by incorporating all ancestral motifs in Thompson's *Motif Index*. We only take into account non-terminal nodes with at least two children. The top-level categories in the index miss an overarching root node, which we add to the tree. Similar as before, we exclude all motifs from the experiment that exhibit maximal mutual information towards each other. This results in 410 possible motifs in the Dutch dataset and 293 motifs in the Frisian dataset.

Table 4 shows the evaluation of the cluster solutions. Interestingly, whereas in the previous evaluation L-LDA and BDM gave similar results, here L-LDA seems to suffer considerably from the addition of ancestral nodes to the observed motifs. The cluster solution obtained from BDM outperforms L-LDA by a substantial margin on all evaluation measures. To obtain a better intuition about why BDM performs better than L-LDA in matching its motif distributions to the hierarchy in Thompson's *Motif Index*, we show part of the hierarchical tree in Figure 3. We display for each motif the top words discovered by the two models. The words discovered by BDM are listed in the left column. The right column displays those of L-LDA.

Various interesting observations can be made on the motif representations in the tree. First, intuitively both L-LDA and BDM are able to discover good quality motifs at the leaves of the tree. Take motif J1780 'Things thought to be devils, ghosts etc.' where L-LDA is able to find some either directly or indirectly related words such as *child molester, butchery* and *world war*. BDM provides a good motif representation for J1150: 'Cleverness connected with the giving of evidence' with words such as *fish pot, fox trap* and *money*. All three items function as important pieces of evidence in the court of law in variants of ATU 1381 'The Talkative Wife and the Discovered Treasure'.

Inspecting the tree provides us with two hypotheses about why L-LDA performs much worse on the cluster evaluation than BDM. First, several motifs contain

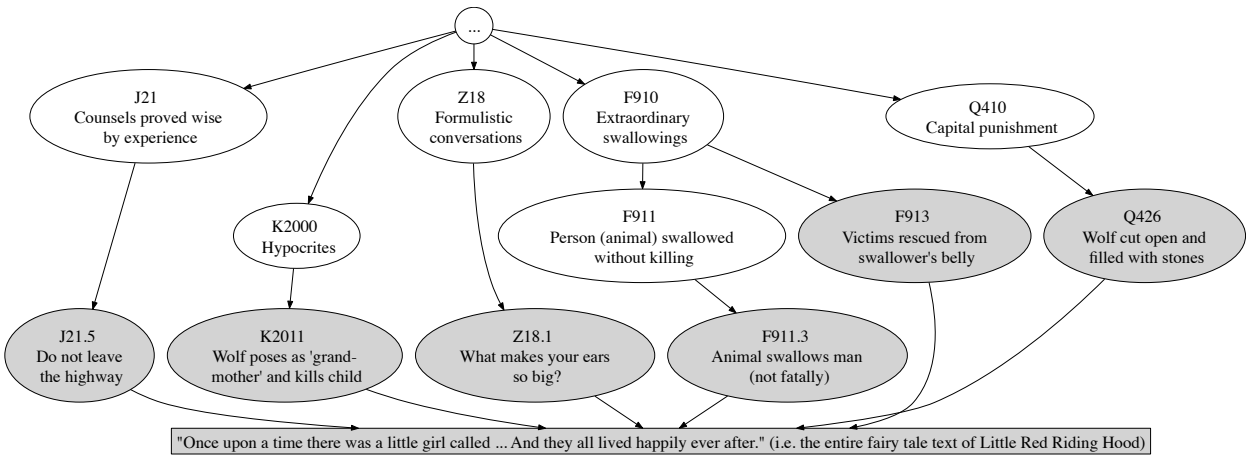


Figure 2. Motif sequence in ATU 333 *Little Red Riding Hood* (gray nodes represent observed motifs), expanded with the ancestor motifs in Thompson's *Motif Index*.

many stop words, although we filtered all words that appear in more than fifty percent of all documents. These content-free words provide little to no clue to discriminate between motif categories, but in L-LDA they play a rather large role in contrast to BDM. The second reason for the superiority of BDM over L-LDA appears to be that BDM incorporates the knowledge from lower-level motifs into the more abstract motifs. A clear example of this is the top-level motif 'J: The wise and the foolish'. Almost all top words are examples of characters in stories that are either wise or foolish. We expanded the original motif set of a document with ancestral motifs. The result of this design choice is that the hierarchical relations between motifs are only implicitly present. Because L-LDA assigns each word in a document to a single motif, motifs that occur in only a few documents will attract more lexically specific words than their ancestors that appear in more documents. This 'restriction' does not apply to BDM, where the same word may be assigned to both lower-level and higher-level motifs. In sum, L-LDA is capable of finding good representations of motifs, but they seem unrelated and the knowledge from higher-level motifs is not inherited by their children.

6. Conclusion

In this paper we applied Labeled LDA to the domain of folktales. We have shown that L-LDA functions as a competitive method to identify motifs in folktales. However, it lags behind on a relatively simple retrieval model that uses Okapi BM25 as its ranking function.

We evaluated the quality of the motifs found by L-LDA and BDM against the most important motif classification system in folktale research. The results showed that both L-LDA and BDM are well capable of discovering high-quality motifs for the lowest-level motifs. However, the motif representation discovered by L-LDA for higher-level motifs are of low quality. In contrast, BDM is able to exploit the hierarchical relations between motifs. The more abstract motifs are in fact generalizations over lower-level ones.

One of the most interesting properties of LDA is that it assigns each word in a document to a single topic. As shown by Ramage et al. (2009), these word-by-word topic assignments could allow us to detect which parts of a text correspond to the tags assigned at the document level. Likewise, we could use this information to localize the specific places at which motifs occur in folktales. Future research should therefore be directed at improving the quality of motif representations as discovered by L-LDA or, in competition with L-LDA, the development of a system that incorporates the motif representations found by BDM, by finding those parts of a text that support a detected motif best.

Acknowledgments

The authors would like to thank the three anonymous reviewers, Theo Meder and Peter van Kranenburg for their helpful comments. The work on which this paper is based has been supported by the Computational Humanities Program of the Royal Netherlands Academy of Arts and Sciences, as part of the Tunes & Tales project.

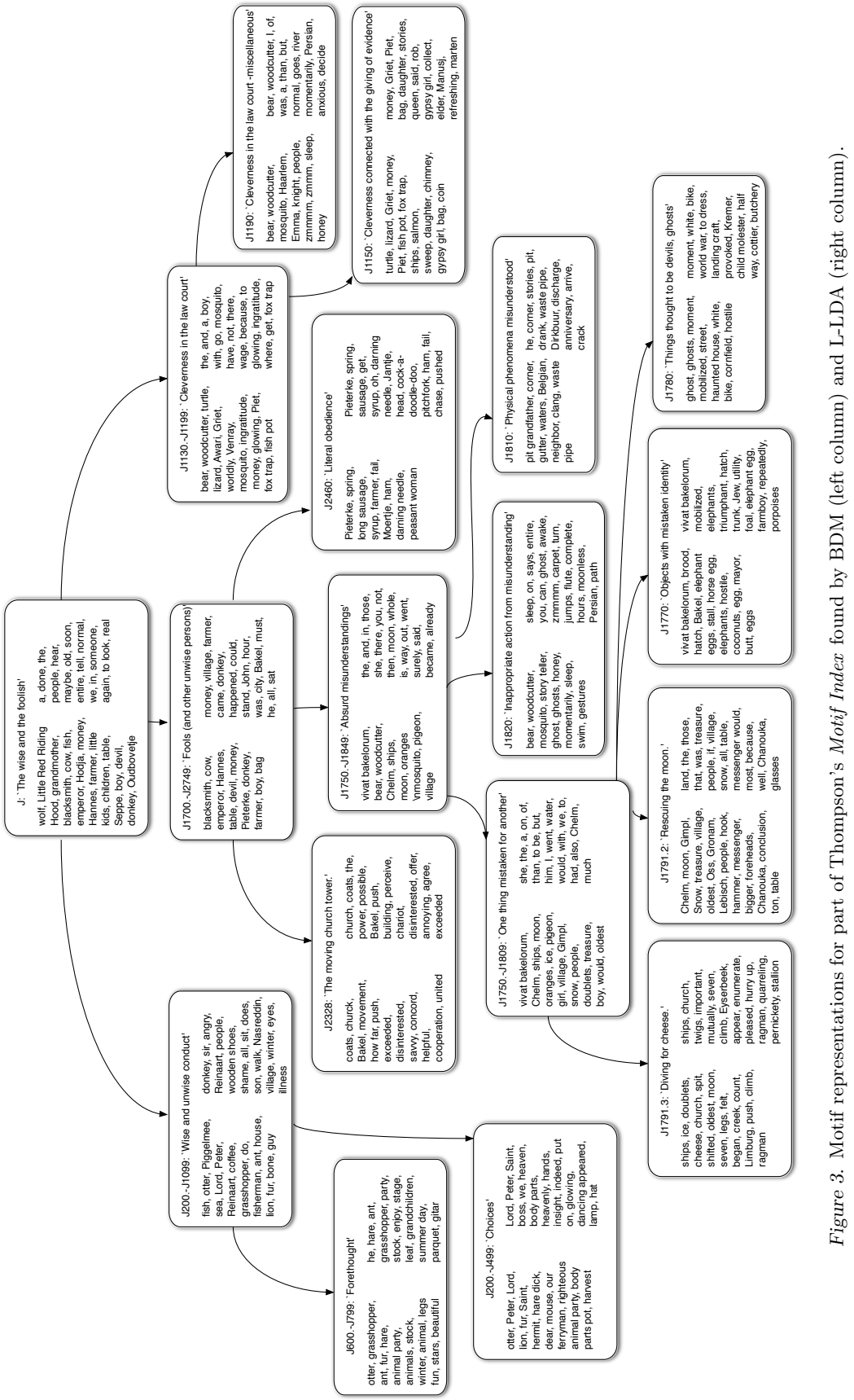


Figure 3. Motif representations for part of Thompson's *Motif Index* found by BDM (left column) and L-LDA (right column).

References

Abello, J., Broadwell, P., & Tangherlini, T. R. (2012). Computational folkloristics. *Commun. ACM*, *55*, 60–70.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, *3*, 993–1022.

Karsdorp, F., Van Kranenburg, P., Meder, T., & Van den Bosch, A. (2012). In search of an appropriate abstraction level for motif annotations. *Proceedings of the 2012 Computational Models of Narrative Workshop* (pp. 22–26). Istanbul, Turkey.

La Barre, K. A., & Tilley, C. L. (2012). The elusive tale: leveraging the study of information seeking and knowledge organization to improve access to and discovery of folktales. *Journal of the American Society for Information Science and Technology*, *63*, 687–701.

Meder, T. (2010). From a dutch folktale database towards an international folktale database. *Fabula*, *51*, 6–22.

Mimno, D., Wallach, H. M., Talley, E., Leenders, M., & McCallum, A. (2011). Optimizing semantic coherence in topic models. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing* (pp. 262–272). Edinburgh, Scotland.

Nguyen, D., Trieschnigg, D., & Theune, M. (2013). Folktale classification using learning to rank. *Advances in Information Retrieval, 35th European Conference on IR Research, ECIR 2013* (pp. 195–206). Moscow, Russia.

Ramage, D., Hall, D., Nallapati, R., & Manning, C. D. (2009). Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing* (pp. 248–256). Singapore.

Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, *3*.

Rosenberg, A., & Hirschberg, J. (2009). V-measure: A conditional entropy-based external cluster evaluation measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (pp. 410–420). Prague.

Rubin, T. N., Chambers, A., Smyth, P., & Steyvers, M. (2012). Statistical topic models for multi-labeled document classification. *Mach Learn*, *88*, 157–208.

Thompson, S. (1946). *The folktale*. New York: Dryden Press.

Thompson, S. (1955–1958). *Motif-index of folk-literature: a classification of narrative elements in folktales, ballads, myths, fables, mediaeval romances, exempla, fabliaux, jestbooks, and local legends*. Indiana University Press.

Tsoumakas, G., & Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, *3*, 1–13.

Uther, H.-J. (2004). *The types of international folktales: a classification and bibliography based on the system of antti aarne and stith thompson*, vol. 1–3 of *FF Communications*. Helsinki: Academia Scientarium Fennica.

Van Gompel, M., Van der Sloot, K., & Van den Bosch, A. (2012). *Ucto: Unicode tokeniser*. Radboud University Nijmegen / Tilburg University. Ilk technical report edition.

Voigt, V., Preminger, M., Ládi, L., & Darány, S. (1999). Automated motif identification in folklore text. *Folklore. An Electronic Journal of Folklore*, *12*, 126–141.

Combining Customer Attribute and Social Network Mining for Prepaid Mobile Churn Prediction

Palupi D. Kusuma PKUSUMA@LIACS.NL
Dejan Radosavljevik DRADOSAV@LIACS.NL
Frank W. Takes FTAKES@LIACS.NL
Peter van der Putten PUTTEN@LIACS.NL
LIACS, Leiden University, P.O. Box 9512, 2300 RA Leiden, the Netherlands

Keywords: churn, call graph, social network mining, spreading activation

Abstract

Customer churn, i.e., losing a customer to the competition, is a major problem in mobile telecommunications. This paper investigates the added value of combining regular tabular data mining with social network mining, leveraging the graph formed by communications between customers. We extend classical tabular churn datasets with predictors derived from social network neighborhoods. We also extend traditional social network spreading activation models with information from classical tabular churn models. Experiments show that in the second approach the combination of tabular and social network mining improves results, but overall the traditional tabular churn models score best.

1. Introduction

Churn, which is defined as the loss of customers to another company, is a crucial problem in the telecommunication industry. As the telecom market has matured and opportunities for growth are limited, retaining existing customers has become a higher priority. In order to minimize the churn rate, mobile telecom players have to form defensive strategies to identify and present the appropriate incentive to subscribers with high churn propensity.

The conventional churn models that exploit traditional predictors, such as demographic information (e.g., age, gender or location), contractual details (e.g., package plan type, contract duration or price), usage facts (e.g., voice call duration, the frequency of sending text-messages) and/or other service-related information (e.g., number of interactions with customer service or number of dropped calls), are typically simple and have a good predictive accuracy (Ferreira et al., 2004; Hadden et al., 2006). However, the predictive accuracy of these models cannot be guaranteed if there is few customer data available, namely in the prepaid segment of the telecommunication industry.

This paper investigates the extent to which social network features derived from the graph formed by communications between customers can be exploited to improve churn prediction accuracy in the prepaid segment. Examples of such features include the number of neighbors of a customer and the number of interactions that a customer has with churned neighbors. This research study was conducted at one of the largest telecom providers in the Netherlands, and a dataset containing 700 million call records was used to assess the quality of the various techniques discussed throughout the paper.

We propose two novel models for churn prediction. The first is a hybrid tabular model, which combines both traditional predictors and social network features to predict churn, aiming to gain significant lift. Logistic Regression and the CHAID algorithm are utilized to derive the tabular models. These churn models, however, do not take into account the influential effect of an individual's decision to his/her social network. A recent work by Dasgupta et al. (2008) has been able to address this problem by constructing a churn model based on a traditional social network mining technique, i.e., spreading activation models. The model propagates the negative churn influence from one subscriber to another in a cascade manner. Besides build-

Appearing in *Proceedings of BENELEARN 2013*.

ing hybrid tabular churn models using a combination of the traditional predictors and the social network features, we also propose a second approach, which extends the traditional propagation model to include the output by traditional churn models.

The rest of the paper is organized as follows. Section 2 presents some related work within the field of churn prediction. Section 3 discusses the call graph and proposed algorithms. The research setup and the empirical models are introduced in Section 4. In Section 5, the experimental results and implications of all scenarios are presented. Finally, Section 6 summarizes the paper and presents some suggestions for future work.

2. Related Work

Churn has been widely analyzed not only in the telecommunication industry (Ferreira et al., 2004; Hadden et al., 2006; Radosavljevik et al., 2010), but also, among others, in the online gaming (Kawale et al., 2009) and banking (Prasad & Madhavi, 2012). Many machine learning techniques, such as decision trees, naive bayes, logistic regression, neural networks and genetic algorithms, are often used to build the tabular churn prediction models.

Ferreira et al. (2004) utilize contractual and demographic information of a Brazilian mobile telecommunication provider to build several postpaid churn models using neural networks, decision trees, genetic algorithms and hierarchical neuro-fuzzy systems. Besides evaluating the predictive power, they also assess the profitability value of those models, claiming that even the churn models with the worst performance are still able to save significant cost in the postpaid segment. Hadden et al. (2006) exploit provisions, complaints and repair interaction data to build the churn models. They claim that the regression tree model performs better than one with neural networks or logistic regression. However, there is no further information regarding the performance comparison between the complaints-based model and the benchmark model based on demographic and contractual variables.

Radosavljevik et al. (2010) investigate the extent to which Customer Experience Management (CEM) data could improve prepaid churn prediction. Several Key Performance Indicators (KPI) of service quality combined with other subscriber data are used to train the decision tree models. Since the CEM data is always available, the constraint on lacking demographic information on the prepaid subscribers could be eliminated. Although the CEM data is predictive, the empirical study shows that there is insignificant gain on this model performance compared to the benchmark.

Several social network studies have been conducted by utilizing mobile call graph data to examine the structure and evolution of social networks (Backstrom et al., 2006; Seshadri et al., 2008), the human mobility patterns (Gyan et al., 2012) and their social interactions (Dasgupta et al., 2008). Dasgupta et al. (2008) analyze the influential impact of the churned neighbors to their social circle by applying a spreading activation-based technique similar to trust metric computations (Ziegler & Lausen, 2004). Using call graph data, they are able to show that churn can be propagated through a social network. Although the study is limited to use social ties information only, reasonable predictive accuracy could still be achieved. The analysis identifies that the churn propensity of a subscriber correlates positively with the number of churned neighbors.

Kawale et al. (2009) conducted a similar study using social network data from a popular online gaming community. They propose a new twist to the existing churn propagation model proposed by Dasgupta et al. (2008) by combining the social influence and user engagement in the game. The user engagement property, which refers to the length of the playing session during the observation period, can be classified as an intrinsic variable. The research shows that the models trained using a combination of social factors and this user engagement property perform better than traditional propagation models. Using collective classification techniques, Oentaryo et al. (2012) are also able to demonstrate that the churn prediction accuracy could substantially be improved by utilizing the combination of traditional user profile and social features.

We apply similar ideas from the above mentioned works. A customer's decision to churn might not only depend on the social influences but also on how they perceive the products and services. On our initial observation, we found that the ratio of the immediate churned neighbors to the number of adjacent neighbors (degree) positively correlates to the churn behavior. When half of the neighbors have churned, the probability of a subscriber to churn is 2 times higher than the baseline churn rate. It implies to some extent that social behavior might have an impact on the subscribers' churning decision. It could be that the hybrid models, which exploit both traditional predictors and social relationships, could outperform the simple social network and the tabular churn model built exclusively using traditional predictors. However, the question is also whether it adds actionable value over existing data. We suspected there may have been some element of publication bias: positive results get published more

often, thus easier to find than non-significant or negative results, at least for trending topics. Hence, we decided to evaluate the business value experimentally.

3. Methodology

A call graph can be derived from raw data of communications between customers. This graph, further discussed in Section 3.1, is essentially a social network which can be leveraged in two ways. Classical 'tabular' models are built on rectangular data sets, one row per customer with subscriber level information. This can be simply extended with attributes (columns) that contain information derived from the social network, as we will outline in Section 3.2. Likewise, a traditional approach to modeling social network dynamics is the spreading activation model, which can be used to model how customer behavior such as churn spreads over the network. Insights from traditional tabular models, more specifically churn scores, can be used to improve these classical social network models, a technique on which we will elaborate in Section 3.3.

3.1. The Call Graph

The *call graph* can be constructed from the Call Detail Records (CDRs) provided by the telecom provider. These CDRs contain detailed facts about mobile interactions, such as source phone number, destination phone number, the type of mobile communication, duration and a timestamp. This information is mapped to a directed social graph $G = (V, E)$ as illustrated in the Figure 1.

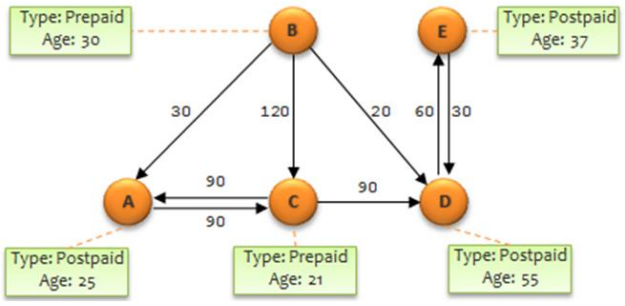


Figure 1. Telecom call graph.

In this call graph, *nodes* denote subscribers and an *edge* represents a mobile interaction between two subscribers. The *edge weight* can be calculated from one variable or a combination of interaction variables, e.g., voice call duration or SMS frequency. It could indicate the interaction intensity or the relationship strength between two nodes. As several interactions could exist between the same pair of nodes, we treat duplicate edges between two nodes as a single edge, by aggregating the weight values. The aggregation method applied in this research is explained in Section 3.3.

3.2. Extended Tabular Churn Models

Many tabular churn models generally exploit either subscriber profile information or social network statistics separately. The predictive power of churn models based merely on the traditional predictors might be reduced in case of many missing values. In our prepaid churn study, we only have an access to limited demographic data because prepaid subscribers are not required to fill in their (accurate) personal information. On the other hand, the social network features might not be predictive enough to influence the churn decision. Neither the traditional models nor the models based exclusively on social networks can cover all aspects of churn on their own. Therefore, we propose to combine both elements to predict churn, adding the features listed in Table 1.

Table 1. Social network features used in the extended tabular churn models.

CATEGORY	VARIABLE
CONNECTIVITY	Count of in/out-degree
	Sum & average of in-/out-weight
	Count & average of voice, SMS & voice+SMS to/from neighbors
	Total and average of edge weight*
	Total interaction frequency with neighbors*
	Total and average frequency with neighbors for voice & SMS separately*
	Degree, 2nd degree & 3rd degree count*
CHURNER CONNECTIVITY	Count of in/out-degree churners
	Sum & average of in/out-weight with churners
	Count & average of voice, SMS & voice+SMS to/from churners
	Total & average edge weight with churners*
	Total interaction frequency with churners*
	Ratio of in/out-degree churners to the total in/out-degree
	Ratio of in/out-weight churners to the total in/out-weight
	Ratio of in/out voice, SMS & voice+SMS frequency with churners to the total in/out-weight
	Ratio of churner weight to the total weight*
	Ratio of interaction frequency with churners to the total interaction frequency*
	Churner degree, 2nd & 3rd degree count*
	Ratio of churner degree to the total degree*
	Ratio of 2nd churner degree to the total 2nd degree*
	Ratio of 3rd churner degree to the total 3rd degree*

*direction is not taken into account

When creating the extended tabular churn models we started with a model based on traditional predictors and added connectivity features from the social network call graph: the in-degree and out-degree, the number of second degree neighbors, sum and average of in-weight and out-weight calculated from duration of voice conversations, SMS and a combination thereof. We also added churn connectivity variables (in-degree and out-degree with churners, etc.), as well as the ratios of the total connectivity measures vs. the churners connectivity measures. A detailed overview of the added social network graph features is presented in Table 1. For a more detailed feature analysis, we refer the reader to Kusuma (2013).

3.3. Extended Social Propagation Models

In this subsection, we discuss an extension of the spreading activation model to measure how churn is diffused around telecom social network (Dasgupta et al., 2008). The churn propagation process begins by initialization of all nodes. In this study, we set the energy of *non-churners* using two different values (see Figure 2). For the *simple propagation approach*, the initial energy of non-churners is set to 0; for the hybrid *extended approach*, it is set to the churn score returned from the regular tabular models.

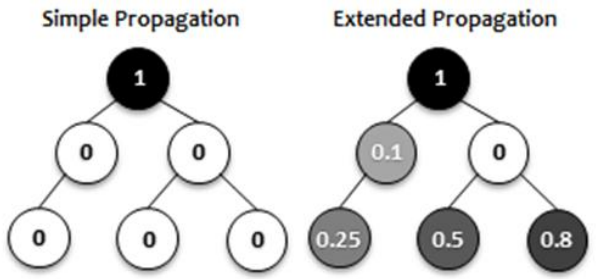


Figure 2. Initial energy of the simple and extended propagation technique.

In the propagation process, for a node $x \in V$, the value of $En(x)$ represents the current amount of energy of a node, and the $En(x, i)$ represents the amount of energy or social influence transmitted to the node x via one or more of its neighbors at stage i (Dasgupta et al., 2008). After energy initialization, a set of previous churners (seeds) is activated. In stage 0, the current energy of the seeds $En(x)$ is used as initial spreading value. Therefore, the current energy value $En(x)$ becomes 0 and amount of energy in a node x at step 0 or $En(x, 0)$ becomes equal to 1.

In each consecutive stage i, the activated nodes transfer a portion of their energy to their neighbors and retain certain portion for themselves. The spreading factor $\delta \in [0, 1]$ controls the proportion of the transmitted energy, denoted by $\delta * En(x, i)$ and the amount of retained energy $(1 - \delta) * En(x, i)$. A spreading factor value of $\delta = 0.8$ means that 80% of the energy is transferred to the neighboring nodes and 20% of the activated energy is retained by the node. This factor value could also be seen as a decay measure because the transferred energy will decline as it gets further away from the source. It implies that the direct neighbors will receive more influence than second degree neighbor and so on. The trust propagation study of Ziegler and Lausen (2004) has shown that people tend to trust individuals trusted by own friends more than individuals trusted only by friends of friends.

Since nodes can have multiple neighbors, the amount of the distributed energy from an active node to each neighbor depends on the tie strengths between the node pair. In Figure 3, for example, the amount of energy transferred from node 1 to node 2 might not be the same as the amount transferred from node 1 to node 3, because the edge weights are not equal.

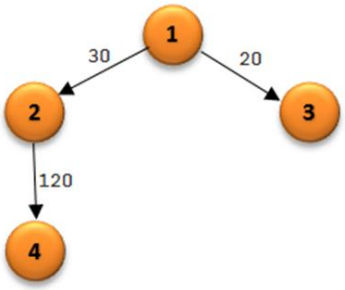


Figure 3. Spreading activation in a weighted graph.

Let y be a neighboring node of an active node x (with $x, y \in V$). We denote the amount of energy transferred from node x to node y in the i-th stage with $En(x, y, i)$. This amount depends on the relative edge weight of the paired nodes. This is determined by a transfer function $f(x, y)$, described in Equation 3 below. The amount of energy transferred is then:

$$En(x, y, i) = \delta * En(x, i) * f(x, y) \quad (1)$$

The amount of energy of node x after the spreading computation is as follows:

$$En(x) = En(x) + (1 - \delta) * En(x, i) \quad (2)$$

There are multiple functions to determine the relative weight between two nodes. The simplest method is using linear edge weight normalization function (Ziegler & Lausen, 2004).

$$f(x, y) = w(x, y) / \sum_{(x,z)} w(x, z) \quad (3)$$

Here, $f(x, y)$ denotes the relative weight of the edge between x and y, $w(x, y)$ represents the weight of that corresponding edge, and $\sum_{(x,z)} w(x, z)$ represents the total weight of all edges connecting node x to its adjacent nodes.

We propagated the churn energy through both a directed and an undirected version of the graph. In the directed graph, energy is propagated only to outgoing edges, and in the undirected graph, both outgoing and incoming edges are used. For churn propagation, the remaining energy after termination ultimately determines the probability of a network member to churn. These churn probability scores are then distributed into score intervals. The upper interval groups contain more subscribers with high churn propensity behavior compared to the lower interval groups. Using the threshold score-based technique, the subscribers/groups with churn scores above a predefined threshold score can each be labeled as a 'churner', and otherwise as a 'non-churners'. As an alternative, a cut-off point can also be determined by specifying the target group size.

4. Experimental Setup

This section describes different specific techniques and assumptions with respect to telecommunications data in Section 4.1, after which the dataset and weighting technique is discussed in Section 4.2. We then give an overview of the seven different scenarios that were used to construct the churn models, outlining our experimental setup in Section 4.3.

For our experiments, we use Chordiant Predictive Analytics Director software to automate variable discretization, variable selection and grouping, to train the scoring models and also to compare the models performance. The default evaluation statistic that is used to measure the performance of the predictors and models is Coefficient of Concordance (CoC). CoC measures the area under the Lorenz curve formed by the percentage of cases with positive behavior against the percentage of cases with negative behavior for each unique score (Harell, 2001).

4.1. Operational Definition of Churn

We constructed models for the prepaid and postpaid telecom segments. Although the definition of churn is different for each segment, we will only discuss the prepaid results because both studies have come to the similar conclusion. Unlike postpaid subscribers, prepaid subscribers are not bound by a contract, which makes it easier for them to churn. Prepaid subscribers need to purchase a credit voucher before using any telecom service. If they do not have sufficient voucher credit, they could not initiate any calls, send SMS/MMS or connect to internet. They could re-enable the service by recharging or topping-up their voucher credit.

A prepaid subscriber is disconnected from the network and he/she is marked as a churner after six consecutive months of inactivity. A prepaid activity could be translated to an outbound voice call, an inbound voice call, an outbound SMS, a data usage or a commercial voucher recharge, also known as top-up. As churn should be detected as early as possible, the disconnection date might not be the appropriate churn date measure (Kraljevic & Gotovac, 2010). The prepaid subscribers might be long gone before they are actually disconnected from the network. Therefore, we define churn as two consecutive months of inactivity. This definition is aligned with many internal studies that are conducted within the company.

4.2. Dataset

We use the CDRs from the whole month of February 2012, which is roughly about 700 million records, to construct the social graph. We include subscribers who have at least one call in February and we base our social network graph on the interactions that occurred in that month. The end goal is to use the traditional predictors as well as the social network information obtained in February, March and April 2012 to predict churn in June 2012. We assume that churn is also a social networking phenomenon, thus subscribers that communicate with people that have churned are more likely to churn themselves. Therefore, we label the nodes/subscribers that churned in the period before May 1, 2012 ('observation 1' in Figure 4) as seeds/churners of the propagation graph explained in Section 4.3. The churn we are trying to predict occurs between May and June 2012 ('observation 2' in Figure 4).

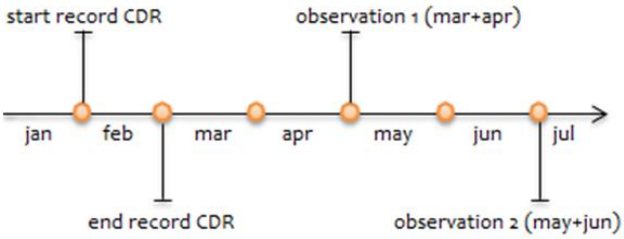


Figure 4. Call Graph Details.

In this research study, we only consider the duration of voice calls in minutes and the count of text messages. We could not explore mobile interactions utilizing the

data connection, i.e., using over the top (OTT) services[1], due to legal issues. Within the company, the postpaid cost of making one minute of a voice call is the same as one SMS. In the prepaid segment, SMS is typically charged roughly half of a minute of voice call. Therefore, we furthermore assume that a text message is equivalent to a voice call of 30 seconds. Hence, we could generalize the edge weight $w(x, y, t)$ between a pair of nodes x and y at time t to include both types of mobile communication, voice calls and SMS and all interactions could all be measured uniformly in seconds. The identifier t represents the hourly timestamp at which the interaction starts, and is ranged from 1 until 29 February 2012.

$$w(x, y, t)' = w(x, y, t) * \begin{cases} 1, & \text{if voice call} \\ 30, & \text{if SMS} \end{cases} \quad (4)$$

Interactions that occurred outside working hours are assigned twice the weight to emphasize their importance. The underlying assumption here is that interactions within working hours mostly indicate communication of professional nature, whereas interactions outside working hours may involve communication of more personal nature (e.g., friends, family), which could have higher influence on the decision to churn. Motahari et al. (2012) shows that members of a family/friends social network are more likely to call each other on the weekend and the engagement ratio value within the family/friends network is at least twice as much compared to the rest of the population. Therefore, we introduce a weight scale $\rho(t)$, which is defined as follows:

$$\rho(t) = \begin{cases} 1, & \text{if t=weekdays (8-17)} \\ 2, & \text{otherwise} \end{cases} \quad (5)$$

$$w(x, y, t)'' = \rho(t) * w(x, y, t)' \quad (6)$$

We also assume that a recent interaction should carry more weight than older ones. Therefore, the daily decay rate $\alpha = 0.2$ is manually selected. The weight value of an edge that is measured on a certain day exponentially decayed according to a predefined rate as follows:

$$w(x, y, t)''' = w(x, y, t)'' * e^{-\alpha * d} \quad (7)$$

Here, the symbol d corresponds to the gap measured in days between the interaction timestamp and the end of the observation period. In our case, d is equal to 28 measured from 1 until 29 February 2012. In the end of the observation period, the weight values are aggregated. As a result, each node pair could only have maximum one edge in each direction, so two edges in total. The equation below formulates the aggregation process of the weight values.

$$w(x, y) = \sum w(x, y, t)''' \quad (8)$$

For an undirected graph, we could simply add up the weights for both directions together as follows:

$$w(x, y) = \sum w(x, y, t)''' + \sum w(y, x, t)''' \quad (9)$$

4.3. Churn Predictive Models

To investigate to which extent social network data could be used to predict churn and possibly could improve churn prediction performance, we trained three tabular data mining models using scoring algorithms and four social network models using a spreading activation algorithm (see Figure 5).

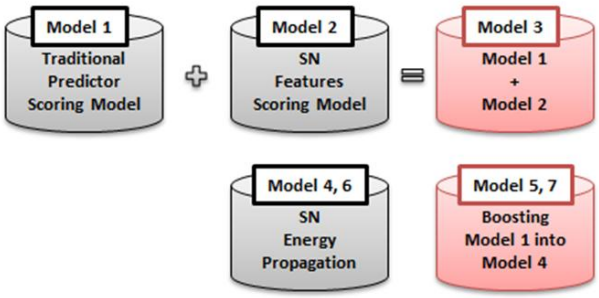


Figure 5. Implementation scenarios.

4.3.1. SCORING MODELS

We apply a logistic regression and a CHAID decision tree algorithm to train our three scoring models:

- Model 1: simple scoring model
- Model 2: social network (SN) scoring model
- Model 3: extended scoring model

Model 1, a simple scoring model, is trained using the traditional churn predictors, using features such as demographic, contractual, handset and usage information. We employ this model as the benchmark model. Model 2 is a social network scoring model, which focuses solely on the social network attributes extracted from call graph, such as the number of incoming and outgoing ties of the first and second degree neighbors. The extended scoring model or the second hybrid model, Model 3, combines the dataset of the first

[1] An over the top service is utilizing the telecom network to perform. However, it does not require any explicit affiliation with the network provider. Examples of over the top application are WhatsApp, Skype or Viber application.

and the second model. This last model is learned from both social network features as well as the traditional churn variables.

4.3.2. PROPAGATION MODELS

The remaining four models are trained using energy propagation techniques based on the previously discussed spreading activation algorithm:

- Model 4: simple propagation model
- Model 5: extended propagation model
- Model 6: simple propagation model undirected
- Model 7: extended propagation model undirected

March and April's churners are used as the source of the energy propagation. Each churned node is given an initial energy of 1. Model 4, which is a simple propagation model, sets the initial energy of non-churners to 0. Model 5 is actually boosting of Model 1 into Model 4. It indirectly incorporates subscribers' intrinsic information into the propagation model. Instead of setting the energy of non-churners to 0, this model assigns the churn score obtained from Model 1 as the initial energy of the non-churner nodes. The intuition behind this idea is that a subscriber might already have a certain tendency to churn due to his/her experience with the provided service. Model 6 and Model 7 are similar to Model 4 and Model 5, except that those models are trained using an undirected instead of a directed graph.

The total energy value that remains after termination is assumed to be the probability of a network member to churn. To study the influential effect of churned neighbors in the social network, we then compare the propensity values of non-churners to the actual known churn class.

5. Results

In this section, the empirical result for each of the implementation scenarios is discussed (see Table 2 and Figure 6). We present and discuss only the scoring models based on decision trees, because these models have a slightly better predictive performance compared to the ones built using logistic regression. Moreover, we only include propagation models with the spreading factor that yield the best prediction results.

Model 3, which is the hybrid model that combines tabular churn predictors and social network variables derived from social network graph, has the highest CoC score on the test set (64.98%). Since it only slightly outperforms Model 1 (64.88%), we can conclude that

Table 2. Coefficient of Concordance of the scoring and propagation models.

	PERFORMANCE ON		
	TRAIN SET	VALIDATION SET	TEST SET
MODEL1	65.48	64.47	64.88
MODEL2	57.93	56.72	56.57
MODEL3	65.65	64.45	64.98
MODEL4	53.34	53.43	53.04
MODEL5	55.26	54.58	55.24
MODEL6	52.07	52.15	52.26
MODEL7	58.39	57.66	58.30

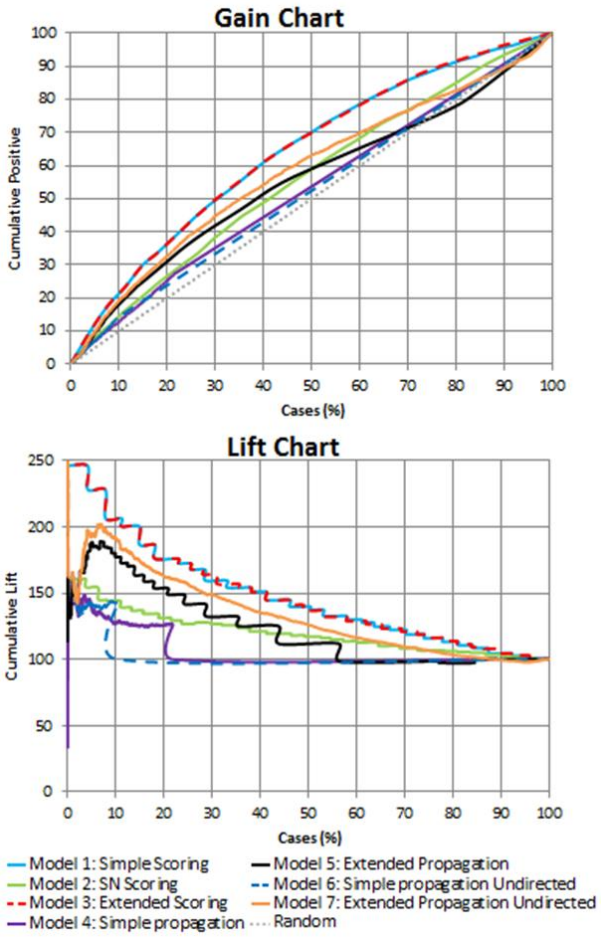


Figure 6. Gain and Lift chart of all models.

adding social network features on top of the traditional churn predictors does not appear to provide any substantial improvement for our scoring model. Model 2 built solely using social network predictors has the lowest predictive accuracy compared to the rest of the scoring models (56.57%). By targeting the top 30% of the subscribers, Model 2 can find only 37% of the

churners, while Model 1 and Model 3 are able to return about 50% of the churners. The lift chart shows that in the top 30% of the cases Model 2 has cumulative lift of 130%, whereas other scoring models have cumulative lift of 160%. In other words, the information derived from the social network is weakly predictive by itself and it fails to outperform the predictive power of the traditional predictors.

As expected, the extended propagation models (Model 5 and Model 7), which incorporate churn scores of the simple scoring model as the initial energy value in the propagation process, outperform the traditional social network propagation models (Model 4 and Model 6). These extended or hybrid models provide better predictive accuracy than the simple propagation models for the directed and the undirected graph. By targeting 30% subscribers, Model 7 is able to correctly predict about 45% churners. It returns 5% less than the tabular churn models, Model 1 and Model 3. Although Model 7 incorporated the traditional predictor elements in the propagation process, the predictive power is still lower than that of the traditional tabular churn scoring models.

The simple propagation models that incorporate only the social neighborhood information, Model 4 and Model 6, have even lower performance compared to Model 2. Unlike Model 2, the simple propagation model uses only the previous churner information within the social network without considering the individual churn propensity. This leads us to believe that the churning behavior of neighbors does not have enough influential effect on other members within a prepaid telecom subscriber social network. Traditional churn predictors apparently have a stronger influence on churn compared to social relationships.

6. Conclusions and Future work

Throughout this paper we have investigated the extent to which social network information can be used to predict telecom churn, and how this information could potentially improve the predictive performance of the conventional churn prediction method. We have assessed the performance of models constructed using the classical tabular data mining, the social network mining and the combination of both mining techniques. The first hybrid model is built by extending the traditional tabular churn predictors with social network variables extracted from the social graph. The second hybrid model is obtained by incorporating results of the traditional tabular churn models to the social propagation graph.

The performance of our models was verified using a large dataset of 700 million call data records. Our initial observation shows that the churn probability is positively aligned with the number of churned neighbors. The regular tabular churn models and the traditional social network models constructed exclusively using social network information score the least. This indicates that social network information alone is not sufficient to predict churn. Overall, the traditional tabular churn models have the best predictive accuracy. The added value of the social network variables to the tabular churn models is rather minimal. Although the second hybrid models are able to outperform the regular propagation models, it still could not beat the performance of the traditional tabular churn models. The contribution of traditional predictors to churn prediction is substantially higher than that of the social network behavior. Moreover, the performance gain of both hybrid models is not substantial enough to justify the computational costs.

The current research study only explores the negative influential effect of previous churners within the social network. Future research could potentially be focused on removing this limitation. The influences from both churners and non-churners could be taken into account, as subscribers might spread messages based on how they perceive the product/service quality. Assuming bad news can have a stronger influential effect than good news, positive influence from non-churners to stay within the network might not be as strong as negative influence from churners. Since our energy propagation model is purely derived from node and neighborhood-based relationships, the spreading activation computations are done locally and subscribers do not have knowledge beyond their direct neighbors. Other algorithms, for example from the field of community detection, are capable to identify the role of subscribers within the social network, such as influencer or adopter. Rather than targeting all future churners, we can minimize our resources by focusing only on churners with high influential power.

References

Backstrom, L., Huttenlocher, D., Kleinberg, J., & Lan, X. (2006). Group formation in large social networks: membership, growth, and evolution. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 44–54.

Dasgupta, K., Singh, R., Viswanathan, B., Chakraborty, D., Mukherjea, S., Nanavati, A. A., & Joshi, A. (2008). Social ties and their relevance to churn in mobile telecom networks. *Proceedings*

of the 11th international conference on Extending database technology, 668–677.

Ferreira, J. B., Vellasco, M., Pacheco, M. A., & Barbosa, C. H. (2004). Data mining techniques on the evaluation of wireless churn. *Proceedings of the European Symposium on Artificial Neural Networks*, 483–488.

Gyan, R., Hui, Z., Zhi-Li, Z., & Jean, B. (2012). Are call detail records biased for sampling human mobility?. *ACM SIGMOBILE Mobile Computing and Communications Review*, *16*, 33–44.

Hadden, J., Tiwari, A., Roy, R., & Ruta., D. (2006). Churn prediction using complaints data. *International Journal of Intelligent Technology*, *13*, 158–163.

Hanneman, R. A., & Riddle, M. (2005). *Introduction to social network methods*. Riverside, CA: University of California, Riverside. `http://faculty.ucr.edu/~hanneman/`.

Harell, F. E. J. (2001). *Regression modeling strategies: With applications to linear models, logistic regression, and survival analysis.* New York: Springer.

Kawale, J., Pal, A., & Srivastava, J. (2009). Churn prediction in mmorpgs: A social influence based approach. *Proceedings of the 2009 International Conference on Computational Science and Engineering*, *4*, 423–428.

Kraljevic, G., & Gotovac, S. (2010). Modeling data mining applications for prediction of prepaid churn in telecommunication services. *Automatika*, *51*, 375–383.

Kusuma, P. D. (2013). Extending traditional telecom churn prediction using social network data. Master's thesis, Leiden University. `http://www.liacs.nl/assets/2013-03PalupiKusuma.pdf`.

Motahari, S., Mengshoel, O. J., Reuther, P., Appala, S., Zoia, L., & Shah, J. (2012). The impact of social affinity on phone calling patterns: Categorizing social ties from call data records. *In Proceedings of the 6th SNA KDD Workshop*, 9–17.

Neo4j. (2012). Neo4j: Community edition (version 1.8.m05) [software]. `http://Neo4j.org/`.

Oentaryo, R. J., Lim, E. P., Lo, D., Zhu, F. D., & Prasetyo, P. (2012). Collective churn prediction in social networks. *In Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 210–214.

Pegasystems. (2008). Predictive analytics director (version cdm 6.3) [software]. `http://www.pega.com/products/decision-management`.

Prasad, U. D., & Madhavi, S. (2012). Prediction of churn behavior of bank customers using data mining tools. *Business Intelligence Journal*, *5*, 96–101.

Radosavljevik, D., van der Putten, P., & Larsen, K. (2010). The impact of experimental setup in prepaid churn prediction for mobile telecommunications: What to predict, for whom and does the customer experience matter?. *Transactions on Machine Learning and Data Mining*, *3*, 80–99.

Seshadri, M., Machiraju, S., Sridharan, A., Bolot, J., Faloutsos, C., & Leskovec, J. (2008). Mobile call graphs: beyond power-law and lognormal distributions. *In Proceedings of the 14th ACM SIGKDD*, 596–604.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data mining: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann. 3rd edition.

Ziegler, C. N., & Lausen, G. (2004). Spreading activation models for trust propagation. *In Proceedings of the IEEE International Conference on e-Technology, e-Commerce, and e-Service*, 83–97.

Recommending Products using Preference Based Modeling

Hui Li HUI.LI@PEGA.COM
Pegasystems Inc. Vinoly building. 8th floor, Claude Debussylaan 20b, 1082MD Amsterdam, The Netherlands

Peter van der Putten PETER.VAN.DER.PUTTEN@PEGA.COM
Pegasystems Inc. Vinoly building. 8th floor, Claude Debussylaan 20b, 1082MD Amsterdam, The Netherlands and LIACS, Leiden University. P.O Box 9512, 2300 RA Leiden, The Netherlands

Maarten Keijzer MAARTEN.KEIJZER@PEGA.COM
Pegasystems Inc. Vinoly building. 8th floor, Claude Debussylaan 20b, 1082MD Amsterdam, The Netherlands

Keywords: product recommendations, online learning, preference modeling, ROC, Naive Bayes

Abstract

Common approaches to product recommendations are offer level propensity models and collaborative filtering. These methods are typically used for respectively a low or a high number of products, and come with specific limitations. In this paper we introduce preference based approaches to product recommendations, aimed at filling the gap between these methods. This approach can be used for any expected value decision problem, with choices across alternatives with common sets of attributes.

1. Introduction

A common way to approach product or offer recommendation problems is to construct a propensity model for every product predicting the likelihood of a positive response. This approach becomes unmanageable when the number of products grows large, changes dynamically at run time or when outcomes are sparse. Alternative approaches are collaborative filtering or recommender systems, but these methods are typically limited to interaction or purchase history, and can't deal with so called 'cold start' situations when a given customer hasn't built up any history yet.

In this paper we present two preference based approaches to this problem. The first method uses explicit models to model customer preferences for specific product attributes, for example a brand or price range. The likelihood to accept an offer is then derived from the likelihood of accepting specific product or offer attributes, given customer, contextual and history data. In the second method simply a single model is used, and product attributes are added to the other inputs for the model. The preference based approach is not limited to product recommendations, but can apply to any decision problem leveraging an expected value framework with a large and dynamic number of alternatives to choose from.

The paper is organized as follows. In section 2 we review relevant background concepts. Section 3 outlines the proposed algorithms, followed by an experimental evaluation in section 4. We end the paper with a discussion (section 5) and a conclusion (section 6).

2. Background

Before presenting our proposed preference based methods in later sections, this section reviews some relevant concepts from decision making under uncertainty as well as the reference approaches for making product recommendations that we are aiming to complement.

2.1. Decision making under uncertainty

From an application point of view the scope of our problem domain is larger than classical ecommerce recommendations. We focus on so called Next Best Action (NBA) systems: engines that provide real time recommendations in all customer interactions, across digital channels such as the web, mobile and social networks, as well as in more physical domains such as contact centers, shops, ATMs etc. These recommenda-

Appearing in *Proceedings of BENELEARN 2013*.

tions may cover more purposes than just recommending products, and there are more factors that go into the decision than just the likelihood of an offer being accepted. In other words, the methods presented in later sections provide a simplified view of both the problem space and approach, but can be further generalized.

The concept of NBA may be new, but a classical framework can be used to model some of the decisions in NBA: expected value, later generalized to expected utility; an early example is Blaise Pascal in 1669 (Pascal, 1995). The idea is that when a choice needs to be made across alternative courses of action, the alternative needs to be chosen with the highest expected value, the summation of likelihood times value for each of the outcomes. For example, to decide between offering A, B or C you choose the offer with the highest likelihood to be accepted times the value (for example profit) on an accept. There are some issues with this approach, outside the scope of this paper, for example it assumes that a decision now does not impact future decisions. In practice it most certainly will; customers and particularly employees will lose trust in these systems if offers are recommended that are high value but low probability to be accepted.

In the scope of this paper the focus is on the likelihood of positive outcomes. Negative outcomes are ignored, as well as value and other business rules, strategies or priorities driving the decision. This can easily be added in real world implementations.

Another relevant decision theoretical concept is multi attribute utility theory (MAUT) (Dyer, 2005). As outlined in the introduction we aim to lift limitations of common propensity modeling and collaborative filtering by modeling customer preferences. The core idea of MAUT is that when choosing from alternatives, once can consider a common set of criteria, with a utility or preference function defined for each criterion. In our preference based methods we use product attributes, either as outcomes (for example predicting the likelihood of accepting an expensive product) or as inputs (predict offer acceptance based on price, in addition to customer and other contextual data), thus modeling preferences for each customer. This can be seen as a special case of a MAUT approach, in which attribute level utility functions are automatically learned.

2.2. Product recommendation algorithms

The preference based methods we will present in the next section onwards are meant to complement other common methods used for product recommendations, such as offer level propensity modeling and collaborative filtering.

In offer level propensity modeling one model is used per product to predict the likelihood of a positive outcome (click, accept, conversion), based on customer data, real time contextual data and past interaction and outcome history. This is then combined with value factors, exclusion rules and other business heuristics to priority rank recommendations. This lends itself well to domains with deep, offer specific decisioning such as financial services, but becomes impractical when there are thousands of products or more. This could lead to an explosion of models, the lack of outcome data for certain products could be an issue and the relationships and similarities between products are not leveraged.

An alternative approach that is often used for recommending across many products (up to hundreds of thousands or millions) is a collaborative filtering recommender system approach. Amazon is one of the classical examples: 'people that bought these books also bought...' This approach has limitations in the sense that only interaction history is leveraged, not necessarily other inputs such as customer characteristics or contextual data. It also suffers from the cold start problem: if there is no interaction history yet for a given customer or history is sparse, it will be hard to make relevant recommendations. We have previously experimented with a hybrid approach that combines customer and offer level modeling and decisioning with collaborative filtering, based on Mahout, which includes a Hadoop and MapReduce based recommender system (Pegasystems, 2012).

The preference based methods are targeting problem domains that are in between these two approaches, i.e. product recommendations for products with thousands to tens of thousands of product instances and single set of product attributes. This can be generalized to any decision problem with similar number of alternatives and a common set of criteria attributes, with automated learning of preferences over these criteria.

3. Preference Based Product Recommendations

In this section we describe the proposed algorithms for preference based product recommendations, which are using an online learning propensity model. We will introduce the online learning method and then show how such a model can be used for composing the two preference based approaches.

3.1. Adaptive models

Our solution to self learning models is a closed-loop system that automates the model creation, deployment, and monitoring process (Walker & Khoshafian, 2012). It is powered by self-learning algorithms that are able to learn from customer feedback in each interaction, therefore establishing customer preferences incrementally while collecting historical data. The adaptive model can use various sources of information about the customer as predictors, such as customer demographics information like age, gender, house, credit, balance, etc. and real time contextual data. It is essentially a scoring model, predicting the likelihood of a positive outcome such as a click on a banner, acceptance of an offer or actual conversion. The adaptive model covers the following steps in the learning and prediction life cycle:

- Data Analysis and pre-Processing: for all candidate predictors, the adaptive model keeps the counts of positive and negative outcomes as sufficient statistics, at a granular level. For numeric predictors this is done at the level of a large number of numeric bins (f.e. age=18-20, 21-23, etc), for nominal predictors for each of most frequent occuring nominal values (f.e. product code = A, B, C etc). We periodically group the numeric bins and nominal values into larger categories, by testing whether there is any significant difference in the likelihood of a positive outcome between bins or nominals. These larger categories will be used by the model, to ensure that the model is not just accurate but also robust, even when it is just starting to learn.

- Sub set feature selection: this is done by analyzing the correlation between candidate predictors. We use an online algorithm to keep track of correlations between candidate predictors, and only the best ones out of each group are used for the model. A correlation threshold can be configured to control the degree of correlations for filtering. Any other real time sub set feature selection method could have been used in this step.

- Model Scoring: adaptive models use a Naive Bayes like technique to combine the selected predictors to generate a score, thus adaptive model generates robust and highly predictive scoring models. The quality measure used for evaluating a scoring model is the Coefficient of Concordance (CoC) (Lin, 1989). Rank concordance methods are preferred because outcome distributions may be highly unbalanced.

- Post-processing: typically uncalibrated scores produced by the scoring model cannot be used directly as probabilities, as underlying model assumptions are rarely met by real world data. Naive Bayes is known to perform surprisingly well for ranking based on model scores, even if the independence assumption of inputs is violated. However, the absolute estimate of probability may suffer in this case (Domingos & Pazzani, 1997; van der Putten & van Someren, 2004), and a proper estimate is key for comparing recommendations properly. The adaptive model thus implements an algorithm to transform model scores to propensities, by binning on the model score range and estimating the likelihood of a positive event for each of the bins. This concludes the adaptive life cycle from the raw predictor inputs to propensities.

3.2. Turning Real-time Contextual Data and Interaction History into Predictors

We have mentioned that adaptive models are able to use customer demographics as input data. During the customer interaction, real-time contextual data can also be taken into account as predictors for the adaptive model. We list a number of potentially predictive data points during a customer interaction:

- Customer intent information such as reason for a call into a contact center (leaving, enquiring, complaining)

- Channel specifics such as the the type of agent that handles the call, browsing history, device used for the interaction

- History about previous recommendations, interactions, outcomes and behavior

In many situations, real time contextual information turns out to be very predictive. The experiments will show how important such attributes can be.

3.3. Preference based methods

As discussed the goal of this paper is to provide preference based approaches, complementing the reference approaches of either using one model per product or using a collaborative filtering recommender system approach.

The first preference based approach is the so-called "*Single Adaptive*" approach. Rather than using one model per product learning from customer and contextual predictors (or leveraging collaborative filtering), only a single model is used for all products in a

product category that also takes product attributes as an input, so that we can scale towards many product instances that can change dynamically at run time.

The second preference based approach is the "*Combined Adaptive*" approach. It is a two-layer method that combines a set of propensity models predicting the likelihood of certain product attributes to be accepted (f.e. price ranges, brands etc), as opposed to products, thus modeling the "preferences" of customers towards certain product features. This approach is elaborated as follows.

We reuse the binning and grouping capabilities in the adaptive model to determine a small number of binned product attributes to focus on as outcomes. For example, the product attribute "Color" has three resulting bins: "Red", "Blue", and "Others". We then create adaptive models to track a user's propensity towards accepting these product attributes (likelihood of accepting a "Red" product etc). Note that the predictors now contain only customer and contextual data, but no product attributes.

Second we combine the propensities of all product attributes to generate a final propensity for each product instance, since one product is uniquely characterized by the set of product attributes. The combination methods can include a simple average over all propensities, or a weighted average in which weights are based on the performance of the adaptive models.

The binning and grouping of predictor values are critical to keep the number of models tractable in this approach. For example, with 10 attributes and 10 possible bins for each attribute, we will define $10 * 10 = 100$ adaptive models. In turn we could in theory model $10^{10} = 1$ *billion* different products, though in practice we will need to score all eligible product instances for a single customer interaction.

4. Empirical Evaluation

In this section we present the empirical evaluation of the proposed approaches for customer behavior prediction, based on an Event Recommendation Dataset. We first describe the data and the experimental setup in detail. Then we further analyze and compare the performance of predictors and algorithms under study.

4.1. Experimental Setup

We use a publicly available dataset from the Event Recommendation Engine Challenge, which is a data mining competition hosted on Kaggle (Kaggle, 2013)[1]. The task is to predict what cultural events users will be interested in based on events they bought tickets for in the past, user demographic information, and what events they have seen and clicked on. The data that we used for experiment include the following information:

- **Interaction data** contains which user is interested in which event. It has the following columns: *userid*, *eventid*, *invited*, *timestamp*, *interested*. "Interested" is a binary variable indicating whether a user has seen and clicked on the event.

- **User data** contains demographic data about users, including the following columns: *userid*, *locale*, *birthyear*, *gender*, *joinedAt*, *location*, and *timezone*.

- **Event data** contains around 110 columns of information about events. They include *eventid*, *starttime*, *city*, *state*, *zip*, *country*, *latitude*, and *longitude*. The other 101 columns are count1, count2, ..., count100, count-other, where countN represents the number of times the Nth most common word stem appears in the name or description of the event. count-other is a count of the rest of the words whose stem was not one of the 100 most common stems.

The interaction data has 15398 records in total, and it contains 2000+ unique users and 8000+ events. We use a 80-20 split into train and test datasets. Stratified sampling on the user's total number of events is performed for splitting.

We use "Coefficient of Concordance" (CoC) (Lin, 1989) to measure the performance of predictors and models. CoC measures how good a model in discriminating good cases from bad cases. It is a value between 0.5 (random distribution) and 1 (perfect discrimination). For binary outcomes, CoC is identical to the area under the receiver operating characteristic (ROC) curve (Fawcett, 2006) (AUC). The AUC is related to the Gini coefficient G by the formula G = 2 * AUC - 1.

We describe in detail the definition of models based on the binning of product features using the event dataset.

Name	Role	Binned Intervals	Grouped Intervals	Grouped Perf...ance
lng	Predictor	201	8	62.56
lat	Predictor	201	8	62.32
city	Predictor	201	4	57.3
country	Predictor	80	4	56.83
c_other	Predictor	200	9	56.63
c_52	Predictor	16	3	55.56
c_3	Predictor	31	4	53.97
c_28	Predictor	12	2	53.56
c_6	Predictor	73	5	53.48
c_8	Predictor	20	3	53.29

Figure 1. The predictive power of event features as predictors in Predictive Analytics Director (PAD). The binned intervals are the initial number of bins (max 200 bins + 1 bin for missing values). The grouped intervals are the number of bins after predictor grouping. Performance is measured using CoC.

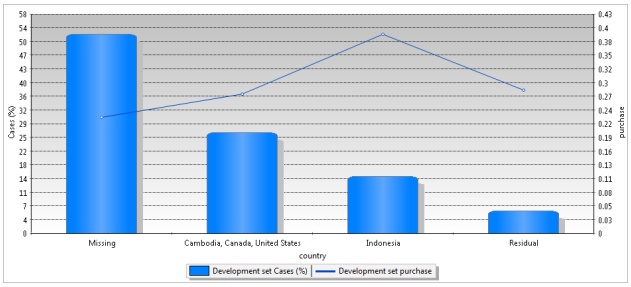


Figure 2. Grouping results for the event feature "Country". From initially 80 countries it is reduced to 4 groups. The percentage of cases in each group and the rate of clicking interest are shown.

4.2. Binning of Product Attributes

As described above, our products (or events), contain 110 features including location information and counts of popular words. If we want to define models on the values of numeric features, naturally we will have to discretize values into ranges or bins. Moreover, it is important to reduce the total number of models in the system, for which we use the predictor binning and grouping capabilities in the Predictive Analytics Director (PAD) tool. Figure 1 shows the top ten predictors ranked by the performance CoC. We can see that the location information such as latitude and longitude of the event, and some keywords (e.g. c-other, c52) are among the most predictive of all event features. After binning and further grouping of bins, the resulting bins of predictor values typically range from 2 to 10. Figure 2 shows one example binning result for feature "country". From initially 80 countries it is reduced to 4 groups. For instance, "Cambodia", "Canada", and "United States" are grouped into one statistically robust bin. Obviously these countries are very different, but from the perspective of accepting offers these are similar. As a result, we define 4 separate adaptive models to track the propensity of each bin. When making a prediction, an event's "country" value will fall into one of the bins, and the propensity of that model will be used as the likelihood for the feature "country".

The binning and grouping capabilities are essential to the Combined Adaptive approach. We choose the top 10 predictors and have in average 5 bins per predictor, therefore around 50 Adaptive models are created. These models can in turn be used to track propensities for a large number of events (e.g. there are 3 million events in the data set).

4.3. Analysis of Predictor Performance

Figure 3 shows the top 15 predictors by performance out of all predictors. We have stated in Section 3 that real-time contextual data gathered during the customer interactions are typically predictive information. In this particular experiment, we have derived a number of attributes from interaction history, for example:

pneg: count of negative responses in previous interactions with the user.

lastoffer_pos: whether the user's response to the last offer is positive or not.

ts_lastoffer: time duration since the last interaction with the user.

We can see that the derived attributes from interaction history are the best performing predictors, with CoC up to 70% (*pneg*). On the other side, the best predictor from user demographics is "locale" (53% CoC).

Figure 3 also shows the predictor grouping results after PAD correlation analysis. The correlated predictors are grouped together and we can choose to use only the best predictor in the group. For example, the set of derived attributes are largely correlated with each other, so we can select *pneg* from this group in the scoring model. There is a parameter called correlation threshold to control the granularity of grouping. Predictor grouping can be used to reduce the total number

[1]The notions of "Products" and "Events" are used exchangeable in this section. The events can be treated as products to be recommended to the users.

Group	Use group	Use p...ictor	Predictors	Grouped...ormance
Group 1	☑	☑	pneg	70.22
		☑	pcount	69.88
		☑	lastpos_id	69.78
		☑	ts_lastpos	69.7
		☑	ppos	69.56
		☑	ts_lastoffer	68.99
		☑	lastoffer_id	68.9
Group 2	☑	☑	lastoffer_pos	63.14
Group 3	☑	☑	lng	62.56
		☑	lat	62.32
Group 4	☑	☑	city	57.3
		☑	country	56.83
Group 5	☑	☑	c_other	56.63
Group 6	☑	☑	c_52	55.56
Group 7	☑	☑	c_3	53.97

Figure 3. Performance of the top 15 predictors and the predictor grouping results by PAD correlation analysis.

Table 1. Performance comparison of Naive Bayes, NBTree, Random Forests (WEKA), and the adaptive model approaches. The model build time is the time elapsed for model training. CoC is calculated on the test set.

Algorithm	CoC	Model Build Time
Naive Bayes	0.729	1.3 sec
NBTree	0.786	322 sec
Single Adaptive	0.783	13.6 sec
Combined Adaptive	0.803	13.6 sec
Random Forests	0.821	6.6 sec

of predictors used in scoring.

4.4. Model Performance Comparison

First we evaluate the performance of Single Adaptive compared to Combined Adaptive approach. Figure 4 shows the ROC analysis of both models. The CoCs are close with 0.783 and 0.803, respectively. When we analyze the ROC curve, we can see that the combined approach outperforms the single model more clearly on the lower-left region of the curve. This is the region of interest: we typically only show a very small number of top recommendations. The improvement by the combined approach is explained by the introduction of partitioning into the multidimensional data space. A manual partition by the product features, together with the reduction of models by binning and grouping, achieves a good balance between model complexity and predictive power.

Second we compare our adaptive approaches with a number of related classifiers in WEKA (Witten et al., 1999), and compare their performance as well as model building time. We can see that the adaptive models outperform the Naive Bayes classifier (with Entropy Discretizer) in WEKA significantly. This can be explained by the advanced data analysis functionality, especially binning, grouping, and correlation analysis in the adaptive model. Interestingly, the Combined Adaptive model based on manual partitioning of product features, outperforms the NBTree (Kohavi, 1996) approach. NBTree is a hybrid algorithm that constructs a decision tree (thus with automatic partitioning) with Naive Bayes classifiers as leaves. Overall, the Random Forests (Breiman & Schapire, 2001) classifier achieves the highest performance among the models under evaluation.

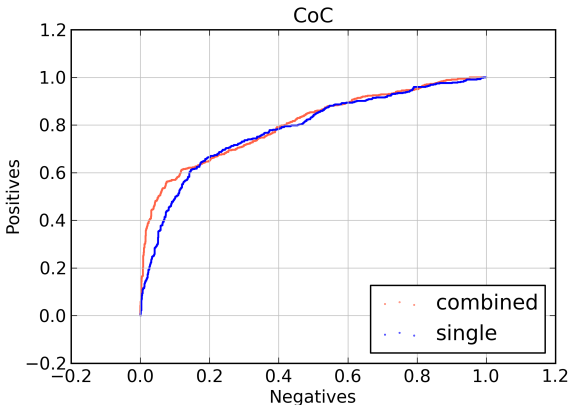


Figure 4. ROC Analysis and Comparison of the Single Adaptive and the Combined Adaptive approach. The ratio of total Negatives vs total Positives is 3 vs 1. CoCs of the Single Adaptive and Combined Adaptive are 0.783 and 0.803, respectively. The Combined approach outperforms clearly on the lower-left region of ROC Curve.

5. Discussions

The advantage of the Single Adaptive approach is its simplicity and it can be fully automated. The Combined Adaptive approach is able to achieve better performance, but requires manual work to define models based on product attributes. Both adaptive approaches learn online after every customer interaction, which is an advantage that enables real-time decisioning. Ensemble learning methods such as Random Forests are robust and perform well, but may require offline training. An interesting research direction is to experiment with online versions of these ensemble learning methods.

6. Conclusions

In this paper we propose a preference based product recommendation approach that is able to recommend across a large number of products. This approach is based on modeling the customer preferences towards product features, and combines individual preferences to form an overall recommendation for the product. The base model is an adaptive model that is able to model customer behavior online. We show that the proposed approach achieves comparable or better performance compared to other methods using a real-life web recommendation dataset. Therefore it establishes a favorable approach towards application domains that scale to thousands of product instances or more and at the same time meeting real-time learning requirements.

References

Breiman, L., & Schapire, E. (2001). Random forests. *Machine Learning* (pp. 5–32).

Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, *29*, 103–130.

Dyer, J. (2005). MAUT multiattribute utility theory. In *Multiple criteria decision analysis: State of the art surveys*, vol. 78 of *International Series in Operations Research and Management Science*, 265–292. Springer New York.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recognition Letters*, *27*, 861–874.

Kaggle (2013). Go from big data to big analytics. `http://www.kaggle.com`.

Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 202–207). AAAI Press.

Lin, L. I.-K. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, *45(1)*, 255–268.

Pascal, B. (1995). *Pensées*. Penguin Classics. English edition translated and introduced by A.J. Krailsheimer. Originally published as Pensées de M. Pascal sur La Religion et sure Quelque Autres Sujets. Guillaume Desprez, Paris, 1669.

Pegasystems (2012). *Hadoop big data support: Gain insights and take actions on big data in real-time* (Technical Report). Pegasystems Inc. Also available at `http://www.pega.com/resources/hadoop-big-data-support`.

van der Putten, P., & van Someren, M. (2004). A bias-variance analysis of a real world learning problem: The CoIL Challenge 2000. *Machine Learning*, *57*, 177–195.

Walker, R., & Khoshafian, S. (2012). *Adaptive bpm for adaptive enterprises* (Technical Report). Pegasystems Inc. See `http://www.pega.com/resources/adaptive-bpm`.

Witten, I. H., Frank, E., Trigg, L., Hall, M., Holmes, G., & Cunningham, S. J. (1999). Weka: Practical machine learning tools and techniques with java implementations.

Modeling Sensor Dependencies between Multiple Sensor Types

Shengfa Miao MIAO@LIACS.NL
Ugo Vespier UVESPIER@LIACS.NL
Joaquin Vanschoren JOAQUIN@LIACS.NL
Arno Knobbe KNOBBE@LIACS.NL
Ricardo Cachucho CACHUCHO@LIACS.NL
Leiden University, The Netherlands
Lanzhou University, China

Keywords: submission information, deadline data, formatting information

Abstract

With the development of sensing and data processing techniques, monitoring physical systems in the field with a sensor network is becoming a feasible option for many domains. When analyzing data collected from the sensor network, there typically exist substantial correlations between various sensor signals. Employing sensors of multiple types will produce a greater signal variation, but sensors will still be sensitive to related aspects of the measured system, that is to say there are certain dependencies. In this paper, we focus on modeling sensors dependencies among sensor types of a sensor network installed on a Dutch highway bridge. This sensor network is composed of three types of sensors: strain gauges, vibration sensors, and temperature sensors. Through linear regression, convolution, envelope and band pass filters, we succeeded in detecting the dependency between strain gauges and temperature sensors in the time domain, and the dependency between strain gauges and vibration sensors in the frequency domain. To gain insight into these dependencies, and how the placement and location of sensors influences them, we further analysed the obtained models in a secondary analysis step. The methods presented in this paper are demonstrated by means of an application on a highway bridge, but we feel that, due to their general nature, they equally apply to other domains amenable to sensing.

Preliminary work. Under review for BENELEARN 2013. Do not distribute.

1. Introduction

With the rapidly decreasing prices for sensors, data gathering hardware and data storage, monitoring physical systems in the field is becoming a viable option for many domains. In fields such as civil engineering, windmills and aviation, so-called Structural Health Monitoring (SHM) systems are becoming popular to understand the actual workings of the system in situ, as well as to monitor the system for any developing faults. More and more, sensor networks consisting of multiple sensor types are being employed in these environments, and large quantities of data are being collected. New methods are required to deal with the proper analysis and interpretation of such data collections. In this paper, we consider a case study of such a multi-sensor network, where non-trivial data processing is required to make sense of the data.

When dealing with multiple sensors measuring a physical system, each individual sensor will be sensitive to some aspects of the system, based on the specific characteristics of the type of sensor and on which part of the system the sensor is placed. This is clearly the case for sensors of different types (such as vibration and temperature sensors), but also for identical sensors attached differently to the system. If two sensors are measuring in each others vicinity, they will likely show some dependency, but in most cases, this dependency will be non-trivial, depending on the location, the orientation and the attachment. As an example, consider an SHM-system employed on an aircraft. In order to measure stresses on a wing, and potential metal fatigue on the wing attachment, *strain gauges* are fitted to the wing attachment. During high-g-force

manoeuvres, the strain gauges will measure high values of strain on the attachment. Other sensors might be placed at the tip of the wing, to measure vibrations caused by turbulence for example. These vibration sensors however, will not be sensitive to sustained bending of the wing, as the sensor simply moves along with the wing, and is only sensitive to rapid changes in the location of the wing. As such, strain gauges are sensitive to different aspects of the dynamics than vibration sensors, although some overlap exists in the physical phenomena captured by either type.

In this paper, we provide some examples of modeling the dependencies between (pairs of) sensors, specifically where multiple sensor types are involved. We will demonstrate the methods on data collected at a Dutch highway bridge within the InfraWatch project (Knobbe et al., 2010; Vespier et al., 2011; Miao et al., 2013). The bridge in question is continually being monitored by a collection of sensors of three different types: strain gauges, vibration sensors, and temperature sensors, all sampling at 100 Hz. One of the main challenges here is to understand the specific focus of each sensor type and to model any relationships across types. Having such a model may help, for instance, to remove certain phenomena measured by one sensor type from the signal of another sensor type. Specifically, we will consider the effect of temperature changes on the strain measurements at various locations on the bridge. As such, we can correct for this temperature effect.

Modeling dependencies between sensors also helps to remove redundancies in the data. Being able to infer the measurements of a particular sensor from the remaining sensor may suggest a smaller, and thus cheaper monitoring set-up. Finally, any modeling over the collection of sensors is beneficial for tracking the health of the bridge over longer periods. Changes in the value of a single sensor will often indicate transient effects, such as traffic or weather, but changes in the *models* of the bridge data indicate structural changes to the actual bridge, warranting further investigation.

A further issue we will be investigating is the effect that location and placement of sensors has on their usefulness within the network. For example, if we wish to understand the effect of temperature on strain measurements, it will be relevant to know where and how these two parameters are being measured. By investigating the dependencies between all pairs of sensors from two types (in this case strain and temperature), we hope to discover practical guidelines for the optimal placement of sensors. In Section 6, we perform a secondary analysis step based on Subgroup Discovery to find key characteristics of sensors in terms of their type, location, mode of attachment and orientation.

2. Preliminaries

In the InfraWatch project, a sensor network with 145 sensors is employed. These sensors are placed along three cross-sections of a single span of the bridge. Each of them is either embedded in the concrete, or attached to the outside of the deck and girders. To measure the strain in different directions on the bridge, we utilize sensors of different types: *vibration sensors* measure vertical motion of the bridge, and *strain sensors* measure horizontal strain caused by deflection of the bridge. In the latter case, we measure strain along both the X-axis and Y-axis. To measure the temperature of different parts of the bridge, we also employ multiple temperature sensors. To formalize this placement, we define each sensor as follows:

Definition 1 (Sensor) *A sensor is a tuple* (t, x, y, e, o)*, where* $t \in \{St, Vi, Te\}$ *indicates the sensor type (strain, vibration, and temperature, respectively),* x *and* y *are its coordinates on the bridge,* $e \in \{embed, attach\}$ *indicates whether the sensor is embedded or attached to the concrete, and* $o \in \{X\text{-}axis, Y\text{-}axis\}$ *indicates the orientation of the sensor.*

In the remainder of this paper, we will make substantial use of linear correlations between two signals. Specifically, we will use the (Pearsons's) correlation coefficient as a measure for how related two signals are, modulo a linear transformation between the two. In many cases, the challenging part is the non-linear operations that will have to be performed to the signals, in order to make them congruent. What remains is a simple linear transformation in order to translate the one scale (for example degrees Celsius) to another (for example strain in $\mu m/m$). Using the correlation coefficient allows us to measure the dependency between two features in a manner that is independent of the scale in which a sensor happens to measure.

3. Strain & Temperature

In this section, we study the relationship between two types of sensor: strain and temperature. The sensor network features a total of 91 strain sensors, 44 of which are embedded, and 47 are attached. Of the 20 temperature sensors, one half is embedded in the surface of the deck, and the other half is attached to the underside of the deck.

Fig. 1 , the absolute correlation coefficients between

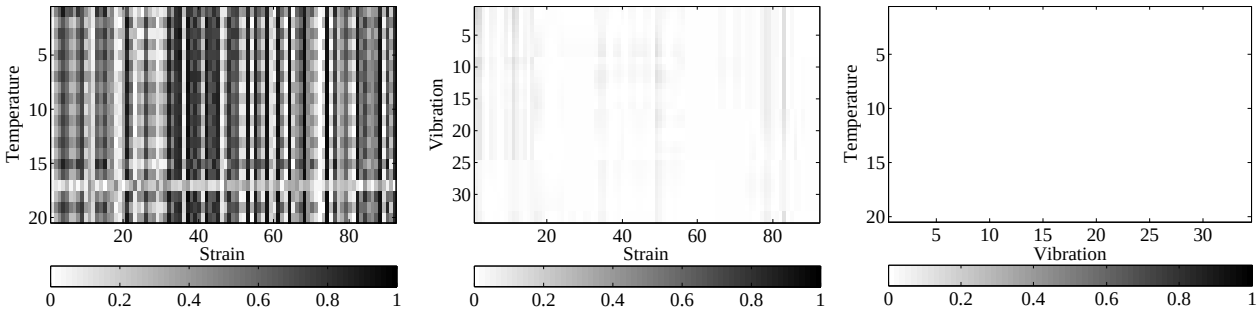


Figure 1. Correlation matrices for St-Te (left), St-Vi (middle) and Vi-Te (right). The numbers on the axes indicate the sensor number.The colorbar value stands for the absolute value of correlation coefficients.

strain and temperature vary from 0 to 0.97. For these sensor pairs with high correlation coefficients, we can simply employ a linear model that assumes the measured strain is directly influenced by the temperature of one of the temperature sensors:

$$S = a \cdot T + b$$

In this model, the coefficients a and b translate between the temperature scale (in Celsius) and the micro-strain scale (in $\mu m/m$). The blue line in Fig. 2 shows the effect of this model applied to a pair of strain and temperature sensor time series that are only moderately related, with $a = -3.288$ and $b = 27.547$ obtained through linear regression over a longer period of time than displayed here. The correlation coefficient for this example is $r = 0.776$, which indicates that the selected pair of sensors are moderately correlated. However, when considering the time series in more detail, one can note that there is a dependency of the strain signal on the temperature measurements, but this relation is non-trivial: it involves a degree of delay: the upward and downward movement of the signal appear to be shifted by several hours.

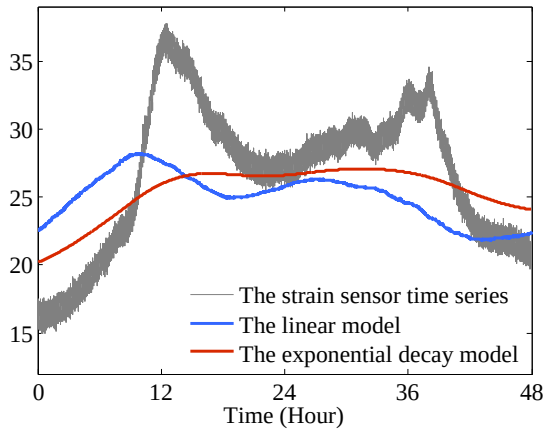


Figure 2. The linear (blue) and exponential decay model (red) between strain and temperature.

The linear model fails to capture the complete effect of temperature on the strain, because the temperature sensor does not actually measure the bridge temperature, but rather the outside temperature. The temperature of the bridge is of course mostly influenced by the outside temperature, but this influence is spread over time, and the bridge temperature will follow changes of outside temperature with a delay. The amount of delay depends on the size and material of the structure, with larger structures (such as the bridge in question) being less sensitive to sudden changes of outside temperature. In other words, a large concrete bridge has a large capacity to store heat, which is mirrored in a slow response of the strain signal.

In the systems analysis field, systems with a capacity are often modeled as a *Linear Time-Invariant* system (Hespanha, 2009). *Time-invariant* indicates that the response of the system does not change over time, which is a reasonable assumption for a bridge (if subtle deterioration of the structure is ignored). LTI systems are *linear* because their 'output' is a linear combination of the 'inputs'. In terms of the bridge, the temperature of the bridge is modeled as a linear combination of the outside temperature over a certain period of time (typically the recent temperature history):

$$T_{bridge}(t) = \sum_{m=0}^{\infty} h(m)T(t-m)$$

where $T_{bridge}(t)$ is the internal temperature and h is an impulse response (to be defined below). Note that this is a special case of *convolution*, a concept that has been extensively studied in signal processing and analysis (Stranneby & Walker, 2004):

$$y(t) = h * x(t) = \sum_{m=-\infty}^{\infty} h(m)x(t-m)$$

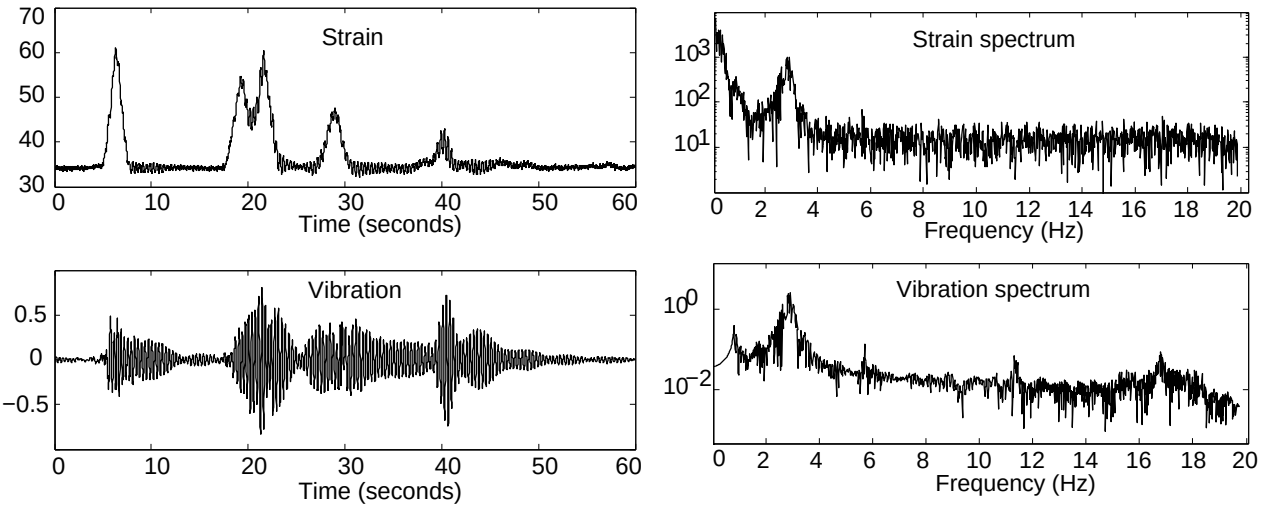


Figure 3. Strain and Vibration signal in the time and frequency domain.

Of the many impulse response functions h, which include for example the well-known moving average operation, we decide to model the delayed effect of the outside temperature using the exponential decay function $h_e(m) = e^{-\lambda m}$ (for $m \geq 0$). In this function, λ is the decay factor, which determines how quickly the effect of past values reduces with time. Note that the resulting equation

$$S = a \cdot h_e * T + b, \text{ where } h_e(m) = e^{-\lambda m} \qquad (1)$$

is the solution to a linear differential equation that is known as *Newton's law of cooling*, which states that the change in temperature of the bridge is proportional to the difference between the temperature of the bridge and its environment:

$$\frac{dT_{bridge}}{dt} = -r \cdot (T_{bridge}(t) - T(t))$$

This is a somewhat simplified representation of reality, in that it assumes that the systems consists of two 'lumps', the bridge and the environment, and that within each lump the distribution of heat is instantaneous. Although in reality this is clearly not the case, it turns out that this model performs fairly well.

For a given pair of sensors and the associated data, we will have to choose optimal values for a, b and λ. It turns out that λ behaves very decently, with only a single optimum for given a and b, such that simple optimisation with a hill-climber will produce the desired result. For Equation 1, we obtain a fitted model for the selected sensor pair shown as the red line in Fig. 2, which clearly demonstrates that the exponential decay model has removed the apparent delay in the data. The fitted coefficients were $a = -12.147, b = 30.463$, and $\lambda = 3 \cdot 10^{-5}$, with a correlation coefficient $r = 0.867$. Considering every possible pair of sensors from St and Te, we find that the correlation coefficients of 47.4% of sensor pairs are improved by the exponential decay model. Indeed, the successful modeling of the dependency for a given pair of sensors still depends on the location and placement of either sensor. In Section 6 we look into the question of finding suitable pairs of sensors in more detail, when we apply Subgroup Discovery to the modeling of St-Te sensor pairs.

4. Strain & Vibration

Our sensor network contains 34 vibration sensors, 15 of which are attached to the bridge deck, while the remaining 19 sensors are attached to the bridge girders. As mentioned in Section 2, both vibration and strain sensors are used to measure the dynamic stresses acting on the bridge. In theory, there should thus be some degree of correlation. However, we failed to detect a strong linear dependency between any pair. As illustrated in Fig. 1 (middle), the correlations between most sensor pairs are quite weak, the highest one for this data being 0.1557. To demonstrate what types of modelling can be done for these two types of sensors, we selected one pair of sensors with a moderate correlation coefficient, as shown in the time domain in Fig. 3 (left). The graphs show that the vibration sensor is a symmetric signal, while the strain sensor time series is not. However, the peaks in both occur consistently, which indicates that they are related. Using a simple correlation, this effect is hidden by the symmetric nature of the vibration signal.

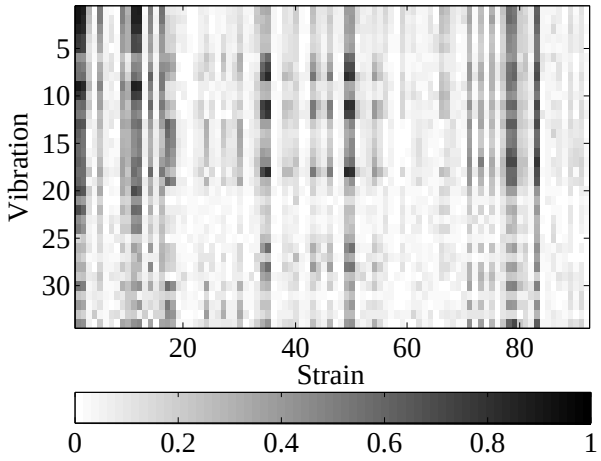


Figure 4. The correlation matrix for St-Vi after applying a band-pass filter in the frequence domain.

Fig. 3, which features the spectra obtained for the two signals by means of a Discrete Fourier Transform (Stranneby & Walker, 2004), shows that despite a lack of a direct relation in the time domain, the signals are actually fairly similar in parts of the spectrum, notably where frequencies above 1 Hz are concerned. Note the big peak around 2.8 Hz in both spectra. In fact, what is missing in the vibration spectrum are the lower frequencies, which correspond to slower bridge movements. In other words, the vibration sensors are not sensitive to gradual changes in the deflection of the bridge, as the sensors themselves simply move along with the bridge. The strain gauges, on the other hand, *are* sensitive even to the slowest changes in bridge deflection. However, both sensors measure shaking of the bridge (frequencies above 1 Hz) in a similar fashion.

Based on these observations, an obvious way to relate St to Vi is to focus on a fairly specific range of frequencies. In our experiments, we have applied a *band-pass filter* to remove all components of the signal outside the range $2.0 - 3.2$ Hz. The linear model between the strain and vibration time series then becomes:

$$BPF_{2-3.2}(S) = a \cdot BPF_{2-3.2}(V) + b$$

in which BPF stands for the band-pass filter operation. After applying the band-pass filter operation to both St and Vi, the correlation coefficient improves from 0.10 to 0.94.

The model we achieved through the band-pass filter operation works well for a small selection of sensor pairs. In Fig. 4, information is displayed on which sensor pairs specifically gain from this operation. Note that some strain gauges correspond well to most of the vibration sensors (dark columns in the matrix). These

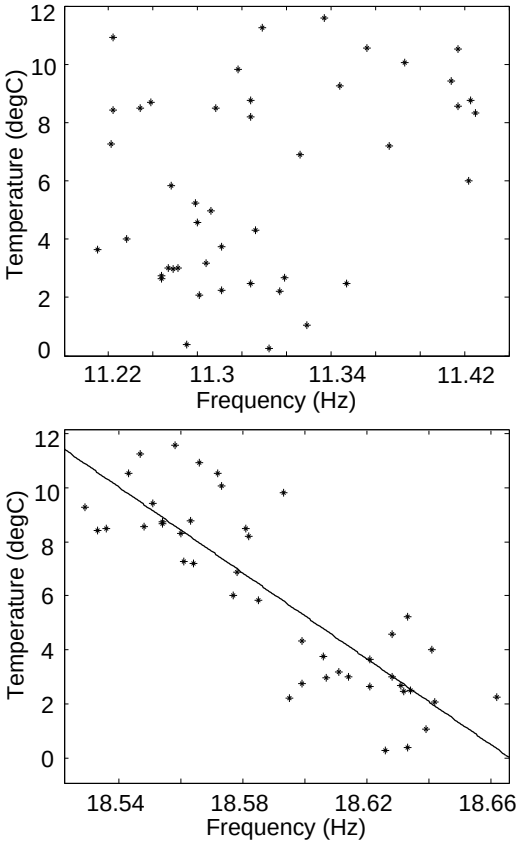


Figure 5. The dependency between modes and temperature.

sensors are primarily located on the right side of the bridge. The few exceptions (St78, St79 and St83) are located on the girder entirely on the other side of the bridge. We look into such observations in more detail in the coming secondary analysis section (Section 6).

5. Vibration & Temperature

As mentioned in the previous section, the vibration spectrum shows little activity in the range below 1 Hz, which happens to be where all of the temperature changes occur (for example due to the daily difference between day and night). For this reason, there are no significant dependencies between the sensors from Vi and Te, shown as Fig. 1 on the right. However, the vibration of the bridge does depend on the temperature. It is well known that bridges tend to oscillate at specific frequencies, and that these frequencies are determined by the stiffness of the structure, which in turn is influenced by changes in the temperature of the material. In a simplified model of a span of the bridge, the *natural frequency* of the span is computed

as follows:

$$f_n = \frac{1}{2\pi}\sqrt{\frac{k}{m}}$$

In this equation, m refers to the mass of the bridge (including the possible load on the bridge), and k is a stiffness coefficient that depends on several factors such as material, humidity, corrosion, etc., but also on temperature. Note that an increasing temperature leads to a decreasing stiffness k, and hence a decrease in frequency, such that we expect a negative relationship between Vi and Te sensors.

The effect of temperature on natural frequencies is widely studied (Song & Dyke, 2006; Xia et al., 2006). After external excitation, for example traffic or wind, a bridge can vibrate in different *modes* (Reynolds & Pavic, 2001). Each mode stands for one way of vibration, which can be vertical, horizontal, torsional or more complicated combinations thereof, and there is one natural frequency corresponding to each. To identify these modes, we use a peak selection method in the spectrum of the vibration sensor (Peeters & De Roeck, 2000). As shown in Fig. 3, we can detect several peaks in the spectrum, each of which is assumed to correspond to a mode. We then consider each mode individually, and look for dependencies between the temperature and the frequency.

In order to consider a substantial range of temperatures, we extracted data from over 45 days, with temperatures between 0 and 12 $^\circ C$. In order to minimize the effect of traffic on m, we selected one hour from each day from 3:00 AM to 4:00 AM. Another motivation for this time-period is the relative stable temperature of both the environment and the bridge. From this hour of data, a spectrum was computed, along with the corresponding modes, as well as the average temperature during this period. Surprisingly, and contrary to many publications (Peeters & De Roeck, 2000; Xia et al., 2006; Liu & Dewolf, 2007; Xia et al., 2011), we find that most modes in the lower ranges of the spectrum (for example the prime one around 2.8 Hz) are not affected by temperature (see Fig. 5 left), at least not in the 12 degrees range available to us. The only mode clearly depending on temperature is around 18.6 Hz, as shown in Fig. 5 right.

6. Analysis of Sensor Properties

As mentioned at the end of Section 3 and 4, we can accurately model some of the strain signals using the temperature signals, and correlate some vibration sensors with strain sensors. However, the models we obtained are not universal for every pair of sensors. To further look into why some sensor pairs work well and others not, we analysed them in a secondary analysis step, where we investigate the influence of various sensor properties such as their location and orientation. The term *secondary analysis* refers to the fact that we are taking the combined set of findings from the previous analysis (the search for sensor dependencies), and treating them as a new data mining task, which is aimed at finding properties of the sensor pairs that help understand why some pairs are easier to model than others. The term *meta-learning* could also apply to this activity.

Our method of choice for this analysis is Subgroup Discovery (SD), which is a descriptive pattern mining technique that aims to outline specific subsets of the data that show a significant deviation of the target, compared to the entire dataset. Our target in this case is the quality of the individual models for sensor pairs (expressed in terms of correlation coefficient), which makes this a regression task. As quality measure for subgroups with a regression target, we use the so-called *(standardized) z-score*, which essentially measures how many standard deviations a subgroup is away from the mean of the entire dataset (see (Pieters et al., 2010) for an overview of quality measures for regression SD). The software we used to conduct these experiment is called Cortana[1], which is a generic toolbox for Subgroup Discovery tasks, including the regression setting that is required here (Meeng & Knobbe, 2011). Any alternative tool for discovering patterns in numeric/nominal data in a regression setting, such as regression trees, would have worked equally well.

Table 1 shows the structure of our data obtained after the initial modeling of sensor pairs. We represent each sensor pair and their properties, including the correlation of the best model, in one row. In the St-Te model we have $91 \cdot 20 = 1820$ rows, and $91 \cdot 34 = 3094$ rows for the St-Vi model. The sensor locations are represented using x and y-coordinates, but in order to allow the SD algorithm to also discover more high-level, interpretable properties, we also introduced several intervals in both dimensions (such as *girder* and *deck* for the y-axis). Additionally, we provided the orientation and type of embedding as nominal attributes.

In our SD run, we search for interesting subgroups with descriptions consisting of conditions on one or more attributes. Although very specific descriptions can be mined, it turns out that fairly simple descriptions are the most informative, so we mine for subgroups of at

[1] It can be downloaded from datamining.liacs.nl/cortana.html, and is also available as a plugin for the KNIME package.

Table 1. Example of the data that was used in the secondary analysis.

Strain								Temperature							
sensor	x	y	embed.	orient.	lane	layer	struct.	sensor	x	y	embed.	lane	layer	struct.	corr.
St1	14	0	attach	X-axis	right	girder	girder	Te1	13	7	embed	right	top	deck	0.139
St1	14	0	attach	X-axis	right	girder	girder	Te2	13	5	attach	right	bottom	deck	0.024
St1	14	0	attach	X-axis	right	girder	girder	Te3	9	7	embed	middle	top	deck	0.068
...															
St2	14	2	attach	X-axis	right	girder	girder	Te1	13	7	embed	right	top	deck	0.277
St2	14	2	attach	X-axis	right	girder	girder	Te2	13	5	attach	right	bottom	deck	0.472
...															

Table 2. The $d \leq 2$ results for the St-Te models ($\mu_0 = 0.533$).

Subgroup description	Coverage %	z-score	μ_{S_i}
St vertical = inside deck & St horizontal ≤ 7	11.0	18.2	0.89
St vertical = inside deck & St orientation = Y-axis	9.9	17.8	0.90
St vertical = inside deck	16.5	16.1	0.79
St vertical = inside deck & Te horizontal ≤ 9	13.2	15.9	0.82
St vertical = inside deck & Te horizontal ≥ 5	13.2	14.1	0.79
St vertical = inside deck & Te embedding = attach	8.5	12.2	0.81
St vertical = inside deck & Te horizontal ≤ 5	6.6	11.2	0.82
St vertical = inside deck & Te embedding = embed	8.2	10.6	0.77
St embedding = embed	47.3	10.3	0.63

most two conditions ($d \leq 2$). The algorithm searches for high-quality subgroups using a beam search with beam width $w = 100$ (Meeng & Knobbe, 2011). A z-score-ranked list of subgroups is returned, of which we report the top-ranking results. Note that we filter the final ranking by removing logical redundant subgroups. A minimum subgroup size of 2 was used.

St-Te models The secondary analysis of the strain and temperature sensors takes the absolute correlation value of each sensor pair as the primary target. The first 9 subgroups (sets of pairs of St-Te sensors) are shown in Table 2. The average correlation over the entire set of pairs is $\mu_0 = 0.533$. The columns contain the subgroup description, the percentage of sensor pairs within the subgroup (i.e. the fraction of the database covered), the z-score, and the average correlation with the subgroup, respectively.

This table shows 2 subgroups of depth one and 7 subgroups of depth two. First, we note that the quality of the St-Te models seems to rely mostly on properties of the strain sensors, rather than the temperature sensors. Apparently, Te sensors provide fairly stable results, whereas for the St sensors, it really depends on the location whether they can be use reliably. Specifically, sensors inside the deck, oriented horizontally on the left side of the bridge[2], appear to work well. Note that such observations are highly useful for the design of future sensor networks, as it provides guidelines to the effective placement of a small collection of sensors. Although subgroups 4 to 8 provide some information as to the placement of Te sensors, these subgroups are not radically different from 'St vertical = inside deck', and have a slightly lower quality (although sometimes higher μ_{S_i}).

St-Vi models Table 3 presents the top-9 subgroups for the strain and vibration models. The results present a much more balanced picture, with both St and Vi properties being crucial for a reliable model. Clearly, the location of sensors at the girders provides the best results, an observation that is corroborated by civil engineering experts in the project. Note that either side of the bridge is much more useful for St placement, compared to the middle of the bridge. We also identify several individual strain sensors (St1, St11, St83) located on both sides of the bridge that play a useful role in many models they feature in. Note that these selected sensors correspond to the three darkest columns in the correlation matrix in Fig. 1 (right).

7. Conclusion and future work

We have demonstrated the use of a number of key data mining and signal processing techniques to model dependencies among multiple sensor types. We have built a linear model to correlate strain and tempera-

[2]The bridge was under construction during this period, and was not being use symmetrically.

Table 3. The $d \leq 2$ results for the St-Vi models ($\mu_0 = 0.139$).

Subgroup description	Coverage %	z-score	μ_{S_i}
St vertical = girder	17.4	31.3	0.36
Vi vertical = girder & St vertical= girder	10.2	28.0	0.40
St vertical = girder & St horizontal = right	6.5	24.8	0.43
St embedding = attach & St orientation = X-axis	38.0	17.7	0.21
St vertical = girder & Vi vertical = under deck	7.2	15.3	0.31
sensor = St1 & Vi vertical = girder	0.6	12.8	0.62
St vertical = girder & St horizontal = left	6.5	12.2	0.28
sensor = St83 & Vi vertical = girder	0.6	11.4	0.56
sensor = St11 & Vi horizontal = right	0.4	11.4	0.68

ture readings, and improved this model through convolution with an exponential response function. In the frequency domain, we used band-pass filters to detect the correlated spectra between strain and vibration sensor time series. For modeling dependencies between vibration and temperature sensor time series, the modes of the spectrum were identified. We note that most low frequency modes are affected little by temperature changes. Finally, we conducted secondary analysis of the models obtained in Section 3 and 4, and extracted subgroups to explain the effects of sensor placement. The extracted rules can be used as guidelines for designing more (cost-)effective networks on future Structural Health Monitoring installations.

References

Hespanha, J. (2009). *Linear system theory.* Princeton university press.

Knobbe, A., Blockeel, H., Koopman, A., Calders, T., Obladen, B., Bosma, C., Galenkamp, H., Koenders, E., & Kok, J. (2010). InfraWatch: Data management of large systems for monitoring infrastructural performance. *Proceedings of Intelligent Data Analysis* (pp. 91–102).

Liu, C., & Dewolf, J. T. (2007). Effect of temperature on modal variability of a curved concrete bridge under ambient loads. *Journal of structural engineering, 133*, 1742–1751.

Meeng, M., & Knobbe, A. (2011). Flexible Enrichment with Cortana - Software Demo. *the* 20^{th} *Machine Learning conference of Belgium and The Netherlands.*

Miao, S., Knobbe, A., Koenders, E., & Bosma, C. (2013). Analysis of traffic effects on a dutch highway bridge. *Proceedings of IABSE.*

Peeters, B., & De Roeck, G. (2000). One-year monitoring of the Z24-bridge: environmental effects versus damage events. *Proceedings of IMAC 18, the International Modal Analysis Conference* (pp. 1570–1576).

Pieters, B., Knobbe, A., & Džeroski, S. (2010). Subgroup discovery in ranked data, with an application to gene set enrichment. *Proceedings of PL 2010 at ECML PKDD.*

Reynolds, P., & Pavic, A. (2001). Comparison of forced and ambient vibration measurements on a bridge. *Proceedings of the International Modal Analysis Conference IMAC, 1*, 846–851.

Song, W., & Dyke, S. (2006). Ambient vibration based modal identification of the Emerson bridge considering temperature effects. *The* 4^{th} *world conference on structural control and monitoring.*

Stranneby, D., & Walker, W. (2004). *Digital signal processing and applications.* Elsevier.

Vespier, U., Knobbe, A., Vanschoren, J., Miao, S., Koopman, A., Obladen, B., & Bosma, C. (2011). Traffic Events Modeling for Structural Health Monitoring. *Proceedings of Intelligent Data Analysis.*

Xia, Y., Hao, H., Zanardo, G., & Deeks, A. (2006). Long term vibration monitoring of an RC slab: temperature and humidity effect. *Engineering Structures* (pp. 441–452).

Xia, Y., Weng, S., Su, J., & Xu, Y. (2011). Temperature effect on variation of structural frequencies: from laboratory testing to field monitoring. *The 6th International Workshop on Advanced Smart Materials and Smart Structures Technology.*

Colour-texture analysis of paintings using ICA filter banks

Nanne van Noord Nanne@tilburguniversity.edu
Tilburg University, Warandelaan 2, 5037 AB Tilburg, The Netherlands

Eric Postma E.O.Postma@tilburguniversity.edu
Tilburg University, Warandelaan 2, 5037 AB Tilburg, The Netherlands

Ella Hendriks Hendriks@vangoghmuseum.nl
Van Gogh Museum, Postbus 75366, 1070 AJ Amsterdam, The Netherlands

Keywords: colour-texture analysis, independent component analysis, image segmentation, colour

Abstract

In human and computer vision, the analysis of visual texture is of pivotal relevance for object recognition and scene understanding. The analysis of texture defined by both intensity and colour variations contributes to the automatic classification in satellite images, medical images, or food processing. Biologically motivated studies have proposed Independent Component Analysis (ICA) as a model for the analysis of colour texture in human vision. To aim of this paper is to determine the effectiveness of ICA for colour texture analysis in computer vision. The effectiveness of ICA on the segmentation of paintings is assessed in unsupervised and supervised settings. The results are encouraging and suggest ICA to be a suitable basis for colour texture analysis.

1. Introduction

Both in biological and artificial vision systems, visual texture provides an important cue for the recognition of objects and scenes (Tuceryan & Jain, 1993). Visual texture refers to the visual appearance of, for instance, textile or surfaces. Texture analysis enables the measurement and quantification of visual texture. Two common applications of texture analysis are texture classification and texture segmentation. In texture classification, images or image patches are classified into useful classes. For instance, the texture of pieces of fruit may be classified according to the type of fruit. In texture segmentation images are subdivided into similarly-textured regions. For instance, texture segmentation can be used to automatically subdivide satellite images into different regions. Although texture is traditionally analysed by considering the spatial variation in pixel intensity (grey scale values) (Tuceryan & Jain, 1993), the application examples given underline the fact that most natural textures are defined in terms of both intensity and colour (see, e.g., (Cheng et al., 2001)). Such textures are called colour textures (Drimbarean & Whelan, 2001). In recent years, colour texture analysis became an active research field. The main challenge in this field is to find an effective way to integrate intensity and colour cues in colour texture analysis (Fernandez & Vanrell, 2012; Wang et al., 2011).

The current study is part of a research project that aims to support painting conservators in their assessments of the ageing of paints. Paintings consist of different types of pigment applied to a canvas. A high-resolution digital reproduction of a painting can be considered to form a mixture of colour texture components, corresponding to the various pigments, brush strokes, and the canvas. From a signal- or image-processing perspective, the problem of recovering the components is a so-called blind source separation problem (Kleinsteuber & Shen, 2012). A widely used algorithm for blind source separation is Independent Component Analysis (ICA) (Hyvärinen & Oja, 2000). ICA is concerned with finding an unmixing matrix by which a mixed signal can be decomposed into its source signals.

Wachtler, Lee, & Sejnowski (2001) showed that ICA is a biologically plausible method for the analysis of nat-

Appearing in *Proceedings of BENELEARN 2013*.

ural scenes and that ICA, when applied to colour images, yields integrated representations of intensity and colour information. Such ICA representations may be invoked as filter banks for colour texture analysis (Jenssen & Eltoft, 2003; Chen et al., 2006). While filter banks for image (or texture) segmentation commonly consist of pre-defined Gabor or Wavelet filters (Ni et al., 2013; Fan, 2012; Jain & Farrokhnia, 1991; Wang et al., 2011). ICA-based filter banks have been shown to outperform Gabor filters (Chen et al., 2006).

To aim of this paper is to determine the effectiveness of ICA for colour texture analysis in both an unsupervised and a supervised setting. Unsupervised ICA-based colour texture segmentation has been studied before (Cheng et al., 2003), but the authors report only qualitative results. To the best of our knowledge, we are the first to study ICA-based colour texture analysis in a supervised and quantitative setting.

The outline of the remainder of the papers is as follows; in Section 2 the ICA method is described. Then, in Section 3 the experimental setup is described, detailing the dataset, experiments and evaluation procedure. The experimental results are presented in Section 4 and the discussion of the results and conclusions are given in Section 5.

2. Independent Component Analysis for colour texture analysis

Our method for performing Independent Component Analysis to colour texture analysis consists of two steps: (1) finding the independent components and treat them as filters, and (2) convolving the ICA filters with an image. In this work the fixed-point algorithm FastICA by Hyvärinen and Oja (2000) is used for all ICA operations.

ICA is a variant of factor analysis that directly finds the independent components of any non-Gaussian distribution (Hyvärinen & Oja, 1997).

2.1. Finding the independent components

The underlying assumption of ICA is that a collection of observed signals or vectors $\mathbf{x}$ consist of statistically independent components (Hyvärinen & Oja, 1997). These components are denoted by $\mathbf{s}$, and can be found by means of a linear transformation of the observations $\mathbf{x}$ with a weight matrix $\mathbf{W}$;

$$\mathbf{s} = \mathbf{W}\mathbf{x}. \tag{1}$$

Although the weight matrix $\mathbf{W}$ and the independent components $\mathbf{s}$ are unknown, the mixing model can be rewritten as

$$\mathbf{x} = \mathbf{A}\mathbf{s}, \tag{2}$$

where $\mathbf{A}$ represents the mixing matrix. It is possible to estimate both the mixing matrix $\mathbf{A}$ and the components $\mathbf{s}$ from the observed signals $\mathbf{x}$ using an unsupervised learning procedure. $\mathbf{A}$ is obtained such that its (pseudo)inverse $\mathbf{W}$ multiplied by $\mathbf{x}_i$ is an estimate of $\mathbf{s}_i$.

In this work ICA is applied to obtain a filter bank g_i, consisting of size $D \times D \times 3$ filters. A weight matrix $\mathbf{W}$ is learned for each image I_i by applying ICA to a collection of randomly sampled patches of size $D \times D \times 3$. Each collection consists of a fixed amount of $50,000$ image patches, sampled from its respective image I_i.

The resulting filter bank is constructed by reshaping each row of weight matrix $\mathbf{W}$ into a $D \times D \times 3$ filter.

2.2. Convolving the ICA filters with an image

The obtained filter banks are applied by convolving each image I_i with its respective filter bank g_i as follows;

$$\mathbf{G}_i(x, y, z) = \mathbf{I}_i(x, y, z) \otimes \mathbf{g}_i. \tag{3}$$

The outcome of the convolution can be summed across all dimensions in order to calculate the energy distribution as follows;

$$f_i = \sum_{y=1}^{N} \sum_{x=1}^{M} \sum_{z=1}^{P} \mathbf{G}_i(x, y, z). \tag{4}$$

The energy value describes how strong the interaction of a filter is with the image at a given location. The energy distribution for two similar areas is expected to be more similar than the energy distribution of two dissimilar areas. The resulting feature matrix f_i consists of a $MN \times |g_i|$ matrix, where each row describes the energy distribution across colour channels for a given pixel.

The size of the filter bank depends on how many independent components are chosen. For our experiments we varied the number of independent components depending on the colour space and for comparability to other results.

3. Experimental setup

The experimental evaluation of our ICA-based colour texture analysis method is performed by segmenting a painting into paints and primed-canvas regions. Both

Figure 1. Daubigny's Garden by Van Gogh (1890).

an unsupervised and a supervised setting will be examined.

3.1. Dataset

The dataset is extracted from a digital reproduction of a painting by Van Gogh: *Daubigny's Garden*, a 50.7×50.7 cm painting that was created in 1890. Figure 1 shows a reproduction of *Daubigny's Garden*. The key characteristic of this painting is that the primed-canvas is visible in some areas of the painting. Ground truth on these primed-canvas locations is available in the form of an images-sized mask consisting of a binary label for each pixel of the 8176×6132 image of Daubigny's Garden. The dataset consists of a random selection of $50,000$ $D \times D$ patches ($D = 8$).

The images in I_i consist of two typical areas of size 1004×750, examples shown in Figure 2, cropped from the high-resolution image of Daubigny's Garden.

3.2. Unsupervised setting

We employ two unsupervised methods: k-means clustering (Cheng et al., 2003) and graph-based clustering (Felzenszwalb & Huttenlocher, 2004), see also (Peng et al., 2013) .

We apply k-means clustering to the energy distribution f_i obtained by convolving an image with the filter bank. The value of k determines the extent of the segmented regions (large for low values of k and small for high values of k). Segmenting an image in a few large regions will often result in a high recall, due to many of the ground layer pixels being encapsulated inside the same region. However, large regions are also much more coarse, and will thus contain many unwanted pixels, resulting in a low precision. The benefit of having a few large regions is that it is trivial to manually select the best regions. In our experiments, k was set to 5, 10, and 100.

The graph-based clustering method has three adjustable parameters; σ, t and min. The first parameter σ is used to smooth the image using a Gaussian filter before segmenting it. The second parameter t which is used to determine the scale of observation, a higher value of t will result in larger components. The third parameter min is used during post-processing to enforce a minimum region size, preventing too many small regions. All experiments were performed with the recommended values of 0.8 for σ and 20 for min. The value of t is varied between the recommended value of 500 and the, in preliminary experiments determined to be appropriate, value of 350.

3.3. Supervised setting

Supervised algorithms make use of training data, or labeled data in order to learn a model that describes the data and allows for predictions to be made about unseen samples. A supervised classifier operates by learning a mapping, for each instance, of the features to the provided label. For the experiments conducted the instances are represented by single pixels. For each instance the ground-truth is noted by a binary label, indicating whether or not it is part of the ground layer. The features consist of the energy distribution f_i.

Ideally a classifier can be trained on a subset of the dataset and can be used to accurately segment an entire image. Several supervised classifiers are evaluated using a standard k-fold cross validation approach which divides the data in a training and test set.

3.3.1. Classifiers

Three classifiers were used in order to tackle the problem at hand, namely the MATLAB implementations of the k-nearest neighbour (k-NN), naive Bayes, and RUSBoost classifiers.

k-NN assigns the unseen samples the (majority) label of the k nearest neighbouring samples in the training set (Hastie et al., 2003). For all experiments described in this paper the value of k was kept at 1.

Naive Bayes is a probabilistic classifier that assigns labels to unseen sample by using Bayes rule (Hastie et al., 2003).

RUSBoost is an ensemble learner that employs a weak learner, a tree, and is particularly well-suited for dealing with class imbalance (Seiffert et al., 2010). RUSBoost combines the AdaBoost boosting procedure with random under-sampling in order to overcome class imbalance, results are obtained using 10,

(a) (b)

Figure 2. Image regions I_a (a) and I_b (b) extracted from Daubigny's Garden

50 and 100 learners.

3.3.2. Evaluation

The evaluation of our colour texture analysis method is split into an evaluation procedure for the unsupervised setting and one for the supervised setting.

Unsupervised setting. The unsupervised setting may yield more than two different regions. For instance, k-means classifier results in a segmentation into k different regions, although our task requires only a binary segmentation into canvas and non-canvas regions. To deal with this issue, we include only a subset of the k regions, namely those that contribute to the overall performance. A region contributes to the performance when the number of true positives for a region (r_{tp}) is greater than the number of false positives for that region (r_{fp}) multiplied by ϵ. As such the solution to finding the best combination of all regions r_i, r_{best} becomes

$$r_{best} = \{r|0 < (r_{tp} - r_{fp} * \epsilon)\}. \quad (5)$$

The value of ϵ was optimised in preliminary experiments yielding a value of 0.23.

Supervised setting. Two commonly used measures for evaluating the performance of classification systems in a supervised setting are precision and recall, which are also used to evaluate segmentation performance (Martin et al., 2004). Precision is a measure of the correctness of the classification. Fewer mistakes lead to a higher precision. Recall is a measure of completeness. The precision of a generated segmentation is calculated by dividing the number of pixels correctly classified as ground layer (true positives) by the total number of pixels classified as ground layer (true positive + false positive), $precision = \frac{tp}{tp+fp}$. Recall is calculated by dividing the true positives by the true positives and the number of ground layer pixels that were not classified as such (false negatives), $recall = \frac{tp}{tp+fn}$.

As a joint measure of precision and recall, we use the F measure, which is the (harmonic) mean of precision and recall: $F = 2 \cdot (precision \cdot recall)/(precision + recall)$.

4. Results

We start the presentation of results by showing an example of the filter bank obtained by applying ICA to our dataset. Figure 3 is obtained by reshaping each row of matrix $\mathbf{W}$ into a n-dimensional filter (where n denotes the number of colour channels). Evidently, the filters capture spatial and colour information from the dataset.

4.1. Unsupervised results

The results for the unsupervised experiments are presented in Tables 1 and 2, for Figures 2(a) and 2(b), respectively. The features that served as input for the unsupervised experiments were obtained using a filter bank of 64 filters, learned from RGB patches. For the initial experiments we have chosen an under-complete basis of 64 rather than 192 independent components in order to facilitate a fair comparison with the results obtained on grey scale images.

For the k-means algorithm a larger value of k results in a higher F-score and precision. In the case of I_a the recall does not change much when changing the value of k. While for I_b smaller values of k give a better result. A possible explanation for this can be found in the difference between the image regions. The perfor-

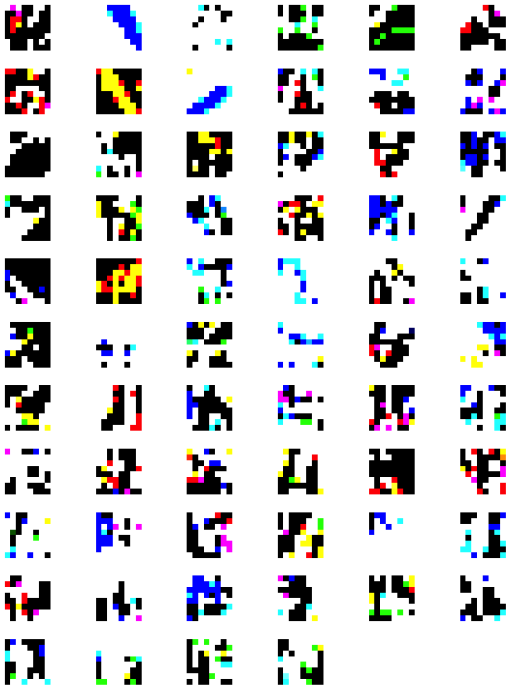


Figure 3. Example of ICA filter bank obtained for $D = 8$.

Table 1. Results of unsupervised algorithms on image I_a

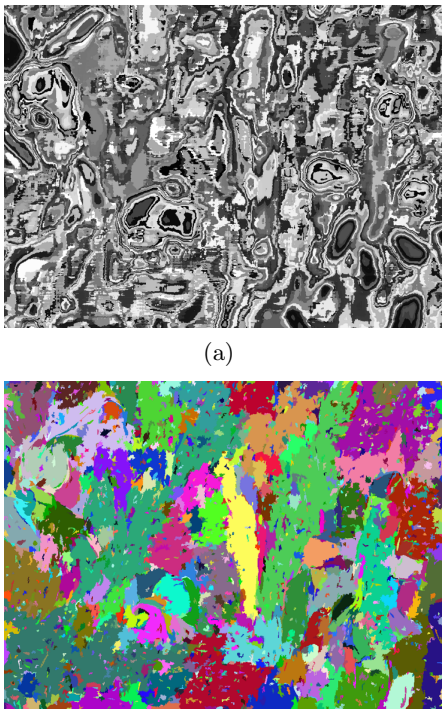
Method	Setting	F-score	precision	recall
k-means	$k = 5$	.309	.230	.469
	$k = 10$	.315	.236	.474
	$k = 100$	.329	.254	.466
graph-based	$t = 350$	.376	.301	.486
	$t = 500$	.349	.303	.413

mance of the graph-based algorithm improves with the lower t value. While the performance with the recommended value ($t = 500$) is on par with k-means (with $k = 100$), the lower value ($t = 350$) results in a better segmentation.

In Figure 4(a) the grey-scale result of the segmentation using k-means with $k = 100$ is shown, which at first glance appears to be more complex than the colour result of the graph-based algorithm shown in Figure 4(b). However, every region in the output of the graph-based algorithm has a randomly-assigned label, which means that identical sections of the image in different locations will be labeled differently. This is not the case for the k-means algorithm that labels identical sections in a consistent way.

Table 2. Results of unsupervised algorithms on image I_b

Method	Setting	F-score	precision	recall
k-means	$k = 5$	.390	.275	.677
	$k = 10$	.393	.280	.660
	$k = 100$	.423	.330	.600
graph-based	$t = 350$	.439	.308	.761
	$t = 500$	.398	.284	.664



(a)

(b)

Figure 4. Segmentation result of Figure 2(a), (a) is the outcome of k-means with $k = 100$, (b) is the outcome of the graph-based algorithm. Each section is indicated by a unique colour.

4.2. Supervised results

The results for the supervised experiments are presented in Table 3 and 4, for Figures 2(a) and 2(b), respectively. As with the results presented in Section 4.1 the results shown in Table 3 and Table 4 are the result of convolutions with filters obtained from the 64 independent components learned from RGB image patches.

The performance of the first classifier, RUSBoost, is about the same for both images. Increasing the number of learners, the main RUSBoost parameter, results in a lower recall, indicating that upscaling the number of of learners will eventually result in lower performance. Similarly to RUSBoost the height of the F-

Table 3. Results of supervised classifiers on image I_a

Classifier	F-score	precision	recall
RUSBoost(10)	.297	.187	.723
RUSBoost(50)	.302	.193	.702
RUSBOOST(100)	.305	.195	.701
Naive Bayes	.281	.174	.730
1-NN	**.556**	.555	.557

Table 4. Results of supervised classifiers on image I_b

Classifier	F-score	precision	recall
RUSBoost(10)	.408	.288	.703
RUSBoost(50)	.419	.308	.652
RUSBoost(100)	.419	.315	.628
Naive Bayes	.378	.251	.766
1-NN	**.532**	.533	.531

score for Naive Bayes stems largely from the high recall, with the precision being very low. Indicating neither classifier is able to accurately distinguish between the different classes and thus assigns the ground-layer label to too large regions.

The precision and recall of the k-NN classifier are very similar, resulting in the highest F-score amongst all classifiers considered. While the recall is lower compared to the other classifiers the higher precision indicates that the k-NN classifier is capable of distinguishing between the majority of the ground-layer pixels and the rest of the painting.

5. Discussion

While the presented results are encouraging we feel that two points are not adequately dealt with, which we address by presenting a selection of preliminary results of our ongoing experiments. The first point concerns the influence of the type of image on the performance. In our current investigations we employ image regions, such as I_c, shown in Figure 5. This image has a stronger distinction between the primed-canvas and paint layers than images I_a and I_b.

The second point is the impact of type of filter bank, colour spaces, and number of features, on segmentation performance. In our studies we focus on new filter banks, such as Log-Gabor filters (Field, 1987), additional colour spaces (e.g., grey-scale and CIELAB), and dimensionality reduction (using PCA).

To provide an impression of how these variations affect the segmentation performance, Table 5 shows the results obtained with the k-NN classifier. The results

Figure 5. Image region I_c extracted from Daubigny's Garden

Table 5. F-scores of the preliminary supervised experiments. Pure refers to results without any PCA preprocessing, PCA(10) are results obtained on the 10 remaining features after PCA.

Filters	Colour space	I_i	Pure	PCA(10)
ICA	Grey	a	.478	.436
		b	.499	.458
		c	.635	.612
	RGB	a	.556	.538
		b	.532	.562
		c	.630	.592
	LAB	a	.528	.519
		b	.565	.510
		c	.646	.638
Log-Gabor	Grey	a	.576	.432
		b	.611	.465
		c	.651	.595
	LAB	a	.572	.595
		b	.617	.465
		c	.652	.596

on image I_c are much better than those on the other two image regions. We suspect that this is due to the greater distinction between the layers present in this image, caused by the increased thickness of the paints applied in this area, completely masking the underlying canvas. The Log-Gabor filters generally perform better than he ICA filters to a degree that depends on the settings. Dimensionality reduction is detrimental for the Log-Gabor filters, but not for ICA.

These preliminary results hint at directions for future study.

6. Conclusion

Our experiments show that ICA in a supervised setting yields better results than in an unsupervised set-

ting. This finding is not surprising, because using the ground truth for modeling textural differences yields a stronger model. More importantly, the k-NN performance obtained with ICA is reasonable and results obtained when analysing colour texture outperform those obtained on grey-scale texture.

While Log-Gabor filters show better results in the setting without PCA, with PCA the ICA filters outperform the Log-Gabor filters. Which is notable as PCA drastically reduces the amount of time required by the classifier. As we extend the method to the entire painting, the size of the dataset increases by a factor of 40, which would put the required time well over what would be feasible for any practical application. Furthermore, there is no difference between the grey scale and colour Log-Gabor settings, which implies that the Log-Gabor method does not efficiently incorporate the additional chromatic information.

Although we have obtained a reasonable performance on a colour texture analysis task using an ICA filter bank, there are two points of concern. First, it is unclear to what extent the complexity of the task influences the performance. A comparable experiment on a painting with a less fine-grained distinction between layers would help in answering this question. Second, the binary labeling of the training data might not have been detailed enough for the classifiers to learn a clear distinction. Further experiments are required to determine whether a richer labeling of the training data would improve performance.

Notwithstanding these concerns, our results are encouraging and suggest ICA to be a suitable basis for colour texture analysis. Future research will focus on exploring ways to improve classification performance by experimenting with conditional random fields, other colour spaces and an extended dataset. We expect that incorporating an interactive user feedback model will improve performance as well as be a practical solution to obtain labeled training data with relatively little effort.

Acknowledgments

The research reported in this paper is performed as part of the REVIGO project, funded by NWO in the context of the Science4Arts research program. The authors thank the anonymous reviewers for their helpful comments.

References

Chen, X.-w., Zeng, X., & van Alphen, D. (2006). Multi-class feature selection for texture classification. *Pattern Recognition Letters*, *27*, 1685–1691.

Cheng, H. D., Jiang, X. H., Sun, Y., & Wang, J. L. (2001). Color image segmentation: Advances and prospects. *Pattern Recognition*, *34*, 2259–2281.

Cheng, J., Chen, Y.-W., Lu, H., & Zeng, X.-Y. (2003). Color- and texture-based image segmentation using local feature analysis approach. *Proc. SPIE 5286, Third International Symposium on Multispectral Image Processing and Pattern Recognition*, 600–604.

Drimbarean, A., & Whelan, P. F. (2001). Experiments in colour texture analysis. *Pattern Recognition Letters*, *22*, 1161–1167.

Fan, J. (2012). Feature learning based multi-scale wavelet analysis for textural image segmentation. In D. Jin and S. Lin (Eds.), *Advances in computer science and information engineering*, vol. 168 of *Advances in Intelligent and Soft Computing*, 461–466. Springer Berlin Heidelberg.

Felzenszwalb, P. F., & Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *International Journal of Computer Vision*, *59*, 167–181.

Fernandez, S. A., & Vanrell, M. (2012). Texton theory revisited: a bag-of-words approach to combine textons. *Pattern Recognition*, *45*, 4312–4325.

Field, D. J. (1987). Relations between the statistics of natural images and the response properties of cortical cells. *J. Opt. Soc. Am. A*, *4*, 2379–2394.

Hastie, T., Tibshirani, R., & Friedman, J. H. (2003). *The Elements of Statistical Learning*. Springer. Corrected edition.

Hyvärinen, A., & Oja, E. (1997). A fast fixed-point algorithm for independent component analysis. *Neural Computing Surveys*, *9*, 1483–1492.

Hyvärinen, A., & Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural Networks*, *13*, 411–430.

Jain, A. K., & Farrokhnia, F. (1991). Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, *24*, 1167–1186.

Jenssen, R., & Eltoft, T. (2003). Independent component analysis for texture segmentation. *Pattern Recognition*, *36*, 2301–2315.

Kleinsteuber, M., & Shen, H. (2012). Blind source separation with compressively sensed linear mixtures. *IEEE Signal Processing Letters*, *19*, 107–110.

Martin, D. R., Fowlkes, C. C., & Malik, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*, 530–549.

Ni, W., Gao, X., & Yan, W. (2013). Sar image segmentation based on gabor filter bank and active contours. In J. Yang, F. Fang and C. Sun (Eds.), *Intelligent science and intelligent data engineering*, vol. 7751 of *Lecture Notes in Computer Science*, 531–538. Springer Berlin Heidelberg.

Peng, B., Zhang, L., & Zhang, D. (2013). A survey of graph theoretical approaches to image segmentation. *Pattern Recognition*, *46*, 1020–1038.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., & Napolitano, A. (2010). Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, *40*, 185–197.

Tuceryan, M., & Jain, A. K. (1993). texture analysis. In C. H. Chen, L. F. Pau and P. S. P. Wang (Eds.), *Handbook of pattern recognition & computer vision*, chapter Texture analysis, 235–276. River Edge, NJ, USA: World Scientific Publishing Co., Inc.

Wachtler, T., won Lee, T., & Sejnowski, T. J. (2001). The chromatic structure of natural scenes. *The Journal of the Optical Society of America A*, *18*, 65–77.

Wang, X.-Y., Wang, T., & Bu, J. (2011). Color image segmentation using pixel wise support vector machine classification. *Pattern Recognition*, *44*, 777–787.

BENELEARN 2013: Abstracts

COSFIRE: A trainable features approach to pattern recognition

George Azzopardi G.AZZOPARDI@RUG.NL
Johann Bernoulli Institute for Mathematics and Computer Science, University of Groningen

Nicolai Petkov N.PETKOV@RUG.NL
Johann Bernoulli Institute for Mathematics and Computer Science, University of Groningen

Keywords: trainable, feature, pattern-recognition, feature learning

In a recent work (Azzopardi & Petkov, 2013), we proposed a trainable features approach to visual pattern recognition. It is called COSFIRE, which stands for Combination of Shifted Filter Responses.

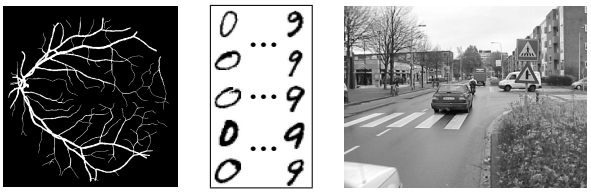
A COSFIRE operator is automatically configured by a specified pattern of interest, referred to as a prototype, and is then able to detect the same and similar patterns in other images. The configuration comprises the determination of the orientations of dominant contour parts and their mutual spatial arrangement.

The output of a COSFIRE operator is computed as the weighted geometric mean of simpler filter responses (e.g. Gabor filters), the properties of which are determined in the configuration stage. COSFIRE operators also achieve tolerance to rotation, scale and reflection.

We demonstrated the effectiveness of the proposed filters in three applications, Fig. 1: the detection of vascular bifurcations in segmented retinal images (DRIVE data set: recall of 97.88% at precision of 96.94%), the recognition of isolated handwritten digits (MNIST data set: 99.48% correct classification), and the detection and recognition of traffic signs in complex scenes (100% recall and 100% precision).

The area of support of a COSFIRE operator is adaptive as it is composed of the support of a number of orientation-selective filters whose relative geometrical arrangement is learned from a given prototype pattern. In contrast, the area of support of other operators, such as SIFT, is typically a square window, the size of which is related to the appropriate scale of the concerned pattern. The presence of noise around a pattern of interest has little or no affect on the output of a COSFIRE operator. Other operators may, however, result in a descriptor that may differ substantially from the descriptor of the same but noiseless pattern.

The proposed COSFIRE operators share similar properties with some shape-selective neurons in visual cortex (Pasupathy & Connor, 1999), which provided inspiration for this work. They are versatile detectors, conceptually simple, easy to implement and are highly effective in practical computer vision applications[1].



(a) (b) (c)

Figure 1. COSFIRE operators are effectively applied to three applications: (a) detection of vascular bifurcations in segmented retinal images, (b) recognition of handwritten digits and (c) spotting of traffic signs in complex scenes.

The COSFIRE approach to feature definition contains an interesting aspect from a machine learning point of view. In traditional machine learning the features to be used are typically predefined and are used to derive other features as (linear) combinations of the original ones with techniques, such as PCA and ICA. With the proposed approach, however, the appropriate prototype features are learned in the configuration of the corresponding COSFIRE operators.

References

Azzopardi, G., & Petkov, N. (2013). Trainable COSFIRE Filters for Keypoint Detection and Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*, 490–503.

Pasupathy, A., & Connor, C. E. (1999). Responses to contour features in macaque area v4. *Journal of Neurophysiology*, *82*, 2490–2502.

[1]Matlab code: http://matlabserver.cs.rug.nl

Data-Adaptive Approximation Selection for Large Time-Series Visualization

Alberto Baggio A.BAGGIO@UMAIL.LEIDENUNIV.NL
Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, Netherlands

Ugo Vespier UVESPIER@LIACS.NL
Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, Netherlands

Arno Knobbe KNOBBE@LIACS.NL
Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, Netherlands

Keywords: Model Selection, Time Series Visualization, Dimensionality Reduction

When dealing with large amounts of data measured from complex time-evolving systems, interactive time series visualization is an effective way to perform exploratory analysis and form an intuition of the system's behavior. Indeed, the human ability to process visual information helps to identify structure and patterns and permits to exploit prior knowledge when applying machine learning and data mining algorithms.

The main challenge when visualizing large time series is to maintain interactivity while allowing the user to quickly zoom in and retrieve detailed portions of the data. Moreover, since the data is plotted in a viewport with a pixel width typically smaller than then the number of points in the time series, some sort of approximation of the original data needs to be performed. The best approximation for this task is the one with the best trade-off between compression ratio and ability to preserve the important perceptual features in the data.

The literature contains many examples of approximation algorithms for time series (Fu, 2011) from frequency-domain methods, such as DFT and the DWT, to time-domain methods such as PAA and APCA. These are however focused on minimizing the Euclidean distance between the original data and the reduced one. There are few algorithms which actually take care of preserving the perceptual features of the original data. Douglas-Peucker (Hao & et al, 2011), PIP (Son & Anh, Oct) and Important Extrema (Fink & Gandhi, 2011) are the most widely cited. However, we show that, while at low compression ratios they model the data pretty well, their L_2 error becomes consistent at high compression ratios.

Considering these limitations, we claim that there is not a single existent technique which behaves well under the interactive visualization requirements depicted above. A good compromise would result from a smart combination of two or more approximations techniques.

We propose a method to select a data-adaptive hybrid approximation obtained by composing diverse techniques. In order to evaluate the reliability of our method, we also define a quality measure for the approximation which keeps into account both the $L2$ error and the capability of preserving important perceptual features. We evaluate our method over 3 months of sensor data (around 500 millions measurements) collected by a sensor network installed on the Hollandse Brug, in the context of the InfraWatch project[1].

References

Fink, E., & Gandhi, H. S. (2011). Compression of time series by extracting major extrema. *J. Exp. Theor. Artif. Intell.*, *23*, 255–270.

Fu, T.-C. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, *24*, 164 – 181.

Hao, M. C., & et al (2011). A visual analytics approach for peak-preserving prediction of large seasonal time series. *Computer Graphics Forum*, *30*, 691–700.

Son, N. T., & Anh, D. T. (Oct.). An improvement of pip for time series dimensionality reduction and its index structure. 47–54.

[1] http://www.infrawatch.com

Asymmetry in Point Set Dissimilarities

Veronika Cheplygina V.CHEPLYGINA@TUDELFT.NL
David M.J. Tax D.M.J.TAX@TUDELFT.NL
Marco Loog M.LOOG@TUDELFT.NL
Pattern Recognition Laboratory, Delft University of Technology, The Netherlands

Keywords: multiple instance learning, dissimilarity representation, non-metric distances

In supervised learning, an object is characterized by a vector of features which aim to distinguish between objects of different classes. In some problems, it may not be straightforward to define what the features should be. This is often the case for complex objects such as graphs, where compressing such objects into a single feature vector representation may increase class overlap.

In multiple instance learing (MIL), the complex objects are called *bags* of *instances*. A bag is a collection of feature vectors, or a point set in a m-dimensional space $B = \{\mathbf{x}_i | i = 1, ..., |B|\} \subset \mathbb{R}^m$. Only bags are labeled $Y(B) \in \{+1, -1\}$, although hidden instance labels $y(\mathbf{x}) \in \{+1, -1\}$ and a mapping $Y(B) = f(\{y(\mathbf{x})\})$ are often assumed. In particular, positive or so-called *concept* instances are assumed to be most important for determining $Y(B)$. This learning setting has originated in drug activity prediction, but has also been applied to classification of images, documents, audio recordings and so forth.

A way of learning with complex objects is to learn from distances or *dissimilarities*, i.e., for MIL by defining a distance measure between bags. Such dissimilarities can be used with the nearest neighbor rule (assigning the object to the class of its closest neighbors), or more generally, as a feature space, where each feature is a dissimilarity to a set of prototypes $\mathcal{R}$(Pękalska & Duin, 2005). In the dissimilarity space (DS) each point set B is represented as a vector $\mathbf{d}(B, \mathcal{R}) = [d(B, R_1), ...d(B, R_{|\mathcal{R}|})]$. In this space, any supervised learner can be used.

Distance measures on complex objects often display non-metric properties, such as asymmetry: $d(B, B') \neq d(B', B)$. Consider the point sets in Fig.1. The metric Hausdorff distance is defined as the overall maximum of the minimum instance distances $\{d(\mathbf{x}, \mathbf{x}') | \mathbf{x} \in B, \mathbf{x}' \in B'\}$ between the two sets. However, we could also measure the minimum, average, etc. leading to possibly non-metric distances. This deteriorates the performance of the nearest neighbor rule, but in the DS, such dissimilarities may be more informative than their metric counterparts.

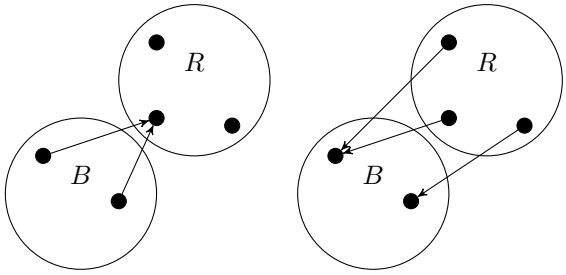


Figure 1. Minimum instance distances between two bags.

In our work(Cheplygina et al., 2012), we show that in some MIL problems, not both directions $d_{to}(B \rightarrow R)$ and $d_{from}(B \leftarrow R)$ are equally informative. In particular, when R is positive, it is often better to use $d_{from}(B \leftarrow R)$ because this ensures that the concept instances in R influence the dissimilarity value, leading to different values for positive and negative bags. On the other hand, with $d_{to}(B \rightarrow R)$ there is a risk that the concept instances in R are disregarded, therefore introducing unnecessary class overlap. In such cases it is not advisable to symmetrize the dissimilarity, but to use the asymmetric versions instead.

References

Cheplygina, V., Tax, D., & Loog, M. (2012). Class-dependent dissimilarity measures for multiple instance learning. *Structural, Syntactic, and Statistical Pattern Recognition*, 602–610.

Pękalska, E., & Duin, R. (2005). *The dissimilarity representation for pattern recognition: foundations and applications*, vol. 64. World Scientific Pub Co Inc.

A Bayesian Approach to Constraint Based Causal Inference

Tom Claassen TOMC@CS.RU.NL
Radboud University, Nijmegen

Tom Heskes TOMH@CS.RU.NL
Radboud University, Nijmegen

Keywords: causal discovery, structure learning, graphical models

We target the problem of accuracy and robustness in causal inference from finite data sets. Our idea is to combine the inherent robustness of Bayesian approaches to causal structure discovery, such as GES, with the theoretical strength and clarity of constraint-based methods such as IC and PC/FCI. We obtain probability estimates on the input statements in a constraint-based procedure, which are then processed in decreasing order of reliability.

Interactions between real-world variables are often modeled in the form of a *causal DAG* $\mathcal{G}_C$. A directed path from A to B in $\mathcal{G}_C$ indicates a **causal relation** $A \Rightarrow B$ in the system. The *causal Markov* and *faithfulness* assumptions link the structure of the graph $\mathcal{G}_C$ to observed probabilistic in/dependencies, which forms the basis behind existing causal discovery procedures.

Method

We break up this inference process into a series of modular steps on basic **logical causal statements** of the form $L : (Z \Rightarrow X) \vee (Z \Rightarrow Y)$, and $L : Z \not\Rightarrow X$. Subsequent statements follow from deduction on the causal properties *transitivity* and *acyclicity*.

We obtain **probability estimates** on logical causal statements by summing the normalized posteriors of all structures $\mathcal{G}$ that entail L through d-separation:

$$p(L|\mathbf{D}) \propto \sum_{\mathcal{G} \in (L)} p(\mathbf{D}|\mathcal{G}) p(\mathcal{G}).$$

Structures over different (small) *subsets* of variables $\mathbf{X} \subset \mathbf{V}$ can already suffice to derive a specific L. This is used in an efficient search strategy over increasing subsets of nodes, where it suffices to keep track of only the *maximum* probabilities obtained so far.

For the likelihood estimates $p(\mathbf{D}|\mathcal{G})$ on possible DAG structures we employ the well-known *Bayesian Dirichlet* (BD) metric.

We still need to account for the fact that the minimal DAG over subset $\mathbf{X} \subset \mathbf{V}$ may be **unfaithful** (uDAG) to the underlying structure. This leads to a modified inference rule, where d-separation remains valid, but the identifiable dependencies are restricted. From this we build a mapping from (possibly unfaithful) uDAGs $\mathcal{G}$ to *valid* logical causal statements $\mathcal{L}$.

Implementation and results

Tests show that a basic implementation of the resulting Bayesian Constraint-based Causal Discovery (BCCD) algorithm already outperforms established procedures such as FCI and Conservative PC. It can also indicate which causal decisions in the output have high reliability and which do not.

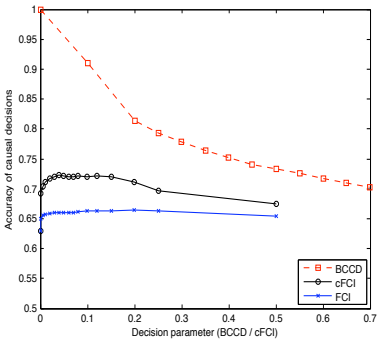


Figure 1. Tunable accuracy of causal decisions in BCCD

The approach is easily adapted into a powerful new independence test that actually *increases* in power for larger conditioning sets. Future extension include scoring MAGs, and allowing for continuous/mixed data.

Acknowledgments

This research was supported by NWO Vici grant nr.639.023.604.

Exceptional Model Mining – Describing Deviations in Datasets

Wouter Duivesteijn WOUTERD@LIACS.NL
Arno Knobbe KNOBBE@LIACS.NL
LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, the Netherlands

Keywords: Local Pattern Mining, Subgroup Discovery, Exceptional Model Mining

Identifying elements that behave differently from the norm in a dataset is a task of paramount importance. Most data mining research in this direction focuses on *detecting* outliers. In Local Pattern Mining, however, we are not just looking for any deviating record or set of records in the data. Instead, we are looking for deviating *subgroups*: coherent subsets that can be *described* in terms of a few conditions on attributes of the data. The existence of such descriptions makes the resulting deviating subgroups more actionable.

'Behaving differently from the norm' can be defined in many ways. Traditionally such exceptionality is measured in terms of frequency (Frequent Itemset Mining), or in terms of a deviating distribution of one designated target attribute (Subgroup Discovery). These concepts do not encompass all forms of deviation we may be interested in. To accomodate a more general form of interestingness, we developed the Exceptional Model Mining framework (Leman et al., 2008; Duivesteijn et al., 2010; Duivesteijn et al., 2012).

The first step of the EMM framework (see Figure 1) is partitioning the attributes in two: one set to *define* subgroups on (the *descriptors*), and one set to *evaluate* the subgroups on (the *targets*). Then a *model class* is selected over the targets, and a *quality measure* over this model class is designed. Finally, the already existing Subgroup Discovery methodology is used to scan the descriptor space for subgroups that perform well according to the quality measure.

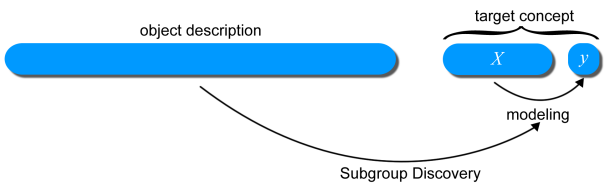


Figure 1. The Exceptional Model Mining Framework

The model class represents interplay between the targets, and the quality measure gauges the exceptionality of model parameters. For instance, we can find subgroups for which two targets are unusually correlated (Leman et al., 2008), subgroups where a classifier performs unusually (Leman et al., 2008), subgroups where a Bayesian network on several nominal targets has a deviating structure (Duivesteijn et al., 2010), and subgroups where a regression model has an exceptional parameter vector (Duivesteijn et al., 2012).

Using EMM instances, we have found subgroups concerning meteorological conditions coinciding with food chain displacement, subgroups defying the economical law of demand, subgroups showcasing the dampening effect of collective bargaining on the distribution of salaries, etcetera. Subgroup significance is tested against a Distribution of False Discoveries, and with the regression model class some subgroups can be pruned without computing the parameter vector.

Acknowledgments

This research is financially supported by the Netherlands Organisation for Scientific Research (NWO) under project number 612.065.822.

References

Duivesteijn, W., Feelders, A., & Knobbe, A. J. (2012). Different slopes for different folks – mining for exceptional regression models with cook's distance. *KDD* (pp. 868–876).

Duivesteijn, W., Knobbe, A. J., Feelders, A., & van Leeuwen, M. (2010). Subgroup discovery meets bayesian networks – an exceptional model mining approach. *ICDM* (pp. 158–167).

Leman, D., Feelders, A., & Knobbe, A. J. (2008). Exceptional model mining. *ECML/PKDD (2)* (pp. 1–16).

Predicting trypsin cleavage sites based on sequence information using decision tree ensembles

Thomas Fannes THOMAS.FANNES@CS.KULEUVEN.BE
Department of Computer Science, Katholieke Universiteit Leuven, Belgium

Elien Vandermarliere ELIEN.VANDERMARLIERE@UGENT.BE
Department of Medical Protein Research, VIB, Ghent, Belgium
Department of Biochemistry, Ghent University, Ghent, Belgium

Leander Schietgat LEANDER.SCHIETGAT@CS.KULEUVEN.BE
Department of Computer Science, Katholieke Universiteit Leuven, Belgium

Lennart Martens LENNART.MARTENS@UGENT.BE
Department of Medical Protein Research, VIB, Ghent, Belgium
Department of Biochemistry, Ghent University, Ghent, Belgium

Jan Ramon JAN.RAMON@CS.KULEUVEN.BE
Department of Computer Science, Katholieke Universiteit Leuven, Belgium

Keywords: decision tree ensemble, proteomics, tryptic cleavage

Proteomics is the large-scale study of proteins, and a typical problem is to identify an unknown protein through mass spectrometry. The protein is cleaved by an enzyme and these peptides are then fed to a mass spectrometer. Afterwards the resulting spectra are compared to in-silico spectra to allow for an identification of the unknown peptides and thus of the unknown protein. Trypsin is the most used enzyme to convert proteins into peptides as it has a high substrate specificity: it cuts exclusively after an arginine and a lysine residue in the protein's sequence.

In our algorithm we propose CP-DT, which is based on a decision tree ensemble and is capable of predicting trypsin cleavage based on the primary structure of a protein and a possible cut position in the sequence. We allow a number of tests on the amino acids type and/or their properties within a window around the possible cut position, e.g. "Is there an amino acid with neutral charge two positions after the cut position?" or "Is there a proline within distance one of the cut position?" We learn a decision tree ensemble where each tree is generated by using a random selection of tests, and the actual prediction is generated by averaging the predicted values of the trees in the forest. The decision tree ensemble is learned by our in-house MIPS framework, a highly-generic, template-based `C++` data mining tool, capable of handling large data streams.

We compare our model with respect to the state-of-the-art "Keil" rules set. CP-DT was learned on a homogeneous dataset retrieved from all 681 193 examples in PRIDE[1]. The model is evaluated on three independent datasets: iPRG (9694 examples), CPTAC (23 842 examples) and MS-Lims (26 079 examples). CP-DT achieves AUROC scores of 84% to 90%, significantly outperforming the Keil rules set with an average improvement in AUROC of 17.9%. In a final step, we use our model to create a database of peptides by applying Naive Bayes, i.e., the probability of cleavage is the product of the start position's cleavage prediction, the end position's cleavage prediction and no cleavage in the middle. This database is compared to typically used databases where each peptide has at most one miscleavage. Here we achieve an AUROC of 93%, outperforming existing techniques with an improvement of 10.0%, which shows a compression of the tryptic search space with respect to traditional databases. We therefore conclude that our trypsin cleavage predictor outperforms the state-of-the-art model.

Acknowledgments

This research has been supported by ERC Starting Grant 240186 MiGraNT: Mining Graphs and Networks: a Theory-based approach.

[1] `http://www.ebi.ac.uk/pride/`

How Well Do Your Facebook Status Updates Express Your Personality?

Golnoosh Farnadi[1,2], **Susana Zoghbi**[2], **Marie-Francine Moens**[2], **Martine De Cock**[1]
{Golnoosh.Farnadi, Martine.DeCock}@UGent.be
{Susana.Zoghbi, Sien.Moens}@cs.kuleuven.be
[1]Department of Applied Mathematics, Computer Science and Statistics, Ghent University
[2]Department of Computer Science, Katholieke Universiteit Leuven

Keywords: Personality recognition, Classification, Online Social Network sites, Facebook

User generated content in online social networking sites provides a potentially rich source of information for applications that rely on personalisation, such as on-line marketing. In this study we contribute to this effort by exploring the use of machine learning (ML) techniques to automatically infer users' personality traits based on their Facebook status updates (i.e., text messages to communicate with friends).

Personality traits are commonly described using five dimensions (known as the Big Five), i.e., extraversion (EXT), agreeableness (AGR), conscientiousness (CON), neuroticism (NEU), and openness (OPN). More than one trait can be present for the same user. We train a binary classifier for each trait that separates the users displaying the trait from those who do not. Formally, given a set of statuses of each user, represented as a feature vector $x \in R^p$, the task is to obtain the set $F = \{(x_1, y_1), ..., (x_l, y_l)\}$, where $y_i \subset C = \{C_1, ..., C_5\}$ corresponding to the five traits. We use a variety of features as input for the classifiers: (1) features related to the text of statuses (e.g., vocabulary and writing style), (2) features about the user's social network (e.g., network size and density) and (3) temporal factors (e.g., frequency of updating status).

Our initial results, based on 250 users and 9917 status updates, show that even with a small set of training examples we can outperform the majority baseline for each trait, with SVM with a linear kernel leading over kNN with k=1 and Naive Bayes. Table 1 presents the results obtained based on accuracy.

	EXT	NEU	AGR	CON	OPN
Majority baseline	0.62	0.55	0.54	0.52	0.70
Classification results	0.68	0.66	0.57	0.55	0.71

Table 1. Classification results based on accuracy

(Golbeck et al., 2011) have recently done a similar study on personality prediction based on all publicly available information in a user's Facebook profile. They obtain promising results on a dataset of 167 users, which is richer than ours in the sense that they have crawled many more profile features (e.g., gender, religion, list of favorite things,...) which were not available to us. Our experiments were carried out on a 250 user sample of a Facebook dataset from the myPersonality project that was released on Feb 1, 2013 (Celli et al., 2013). More efforts on predicting personality traits using the myPersonality project data are undoubtedly underway, but no work has been published yet based exclusively on Facebook status updates, network properties and time factors, like our work.

We also trained the classifiers on a corpus of 2468 essays labeled with personality traits (Mairesse et al., 2007). These essays are on average much longer than the status updates, and the context is different. Still, our results show that models trained on the essay dataset perform well on the Facebook data, and vice versa. This provides evidence that ML based models for personality trait recognition generalise across different domains. Advantages of this are that training examples from different social media platforms can be used in combination to train more accurate models and that such models are also applicable on social network sites for which no training data is available.

References

Celli, F., Pianesi, F., Stillwell, D., & Kosinski, M. (2013). Workshop on computational personality recognition (shared task). *Proc. of WCPR13, in conjuction with ICWSM13.*

Golbeck, J., Robles, C., & Turner, K. (2011). Predicting personality with social media. *Proc. of CHI.*

Mairesse, F., Walker, M. A., Mehl, M. R., & Moore, R. K. (2007). Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research*, *30*, 457–500.

Mutual Information: an Adequate Tool for Feature Selection

Benoît Frénay BENOIT.FRENAY@UCLOUVAIN.BE
Gauthier Doquire GAUTHIER.DOQUIRE@UCLOUVAIN.BE
Michel Verleysen MICHEL.VERLEYSEN@UCLOUVAIN.BE
Machine Learning Group - ICTEAM, Université catholique de Louvain, Place du Levant 3, B-1348 Louvain-la-Neuve

Keywords: mutual information, probability of misclassification, Hellman-Raviv and Fano bounds, MSE, MAE

Because of the curse of dimensionality (Bellman, 1961), it is often necessary to reduce the dimensionality of data before learning. For example, micro-array datasets contain only a few tens of instances with thousands of features. Such preprocessing can be achieved by feature selection.

In the field of machine learning, mutual information (MI) has been widely used as a multivariate criterion of nonlinear feature relevance (Kojadinovic, 2005; Rossi et al., 2006; Doquire & Verleysen, 2011). Indeed, it is well-known in information theory that $I(X;Y)$ measures the reduction of uncertainty about a target Y when a set of features X are observed. However, other criteria are commonly used for classification and regression to assess the quality of models, like e.g. accuracy or mean square error (MSE). This presentation reviews several works (Frénay et al., 2012b; Frénay et al., 2013; Frénay et al., 2012a; Doquire et al., 2013) which address the relationships between MI and these criteria in a feature selection context.

Bounds exist for classification between MI and the probability of error, like e.g. the Hellman-Raviv inequality and the Fano inequalities. These bounds are sometimes used in the literature to justify the use of MI, claiming that selecting the feature subset which maximises MI corresponds to minimising the probability of error. However, this is not necessarily true (Frénay et al., 2012b; Frénay et al., 2013) and it is easy to design counterexamples where a suboptimal feature subset has an higher MI than the optimal feature subset. Hopefully, such failures have limited impact in practice (Frénay et al., 2013; Doquire et al., 2013).

There exists a deterministic link for regression between MSE and MI, as well as the mean absolute error (MAE), when the estimation error has a Gaussian, Laplacian or uniform distribution (Frénay et al., 2012a). In these realistic cases, MI can be safely used to perform feature selection. It is also possible to design counterexamples for regression, like e.g. when the estimation error has a Student distribution with a variable number of degrees of freedom, but the impact of such failures remains limited in practice.

In conclusion, this presentation provides both theoretical and empirical evidences that MI is not a perfect feature selection criterion in all situations, but is still a valuable criterion for feature selection, which is supported by the large number of successful applications in the literature.

Acknowledgments

Gauthier Doquire is funded by a Belgian F.R.I.A. grant.

References

Bellman, R. E. (1961). *Adaptive control processes - a guided tour*. Princeton University Press.

Doquire, G., Frénay, B., & Verleysen, M. (2013). Risk estimation and feature selection. *Proc. ESANN*.

Doquire, G., & Verleysen, M. (2011). Feature selection with mutual information for uncertain data. In *Data warehousing and knowledge discovery*, vol. 6862 of *Lecture Notes in Computer Science*, 330–341.

Frénay, B., Doquire, G., & Verleysen, M. (2012a). Is mutual information adequate for feature selection in regression ? Neural Networks, Submitted.

Frénay, B., Doquire, G., & Verleysen, M. (2012b). On the potential inadequacy of mutual information for feature selection. *Proc. ESANN* (pp. 501–506).

Frénay, B., Doquire, G., & Verleysen, M. (2013). Theoretical and empirical study on the potential inadequacy of mutual information for feature selection in classification. *Neurocomputing, Accepted for publication*.

Kojadinovic, I. (2005). On the use of mutual information in data analysis : an overview. *Proc. ASMDA*, 738–747.

Rossi, F., Lendasse, A., Francois, D., Wertz, V., & Verleysen, M. (2006). Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. *Chemometr. Intell. Lab., 80*, 215–226.

Efficient Feature Selection via Online Co-regularized Algorithm

Sultan Imangaliyev S.IMANGALIYEV@TNO.NL
Evgeni Tsivtsivadze E.TSIVTSIVADZE@TNO.NL
Bart Keijser B.KEIJSER@TNO.NL
MSB Group, The Netherlands Organization for Applied Scientific Research, Zeist, The Netherlands

Wim Crielaard W.CRIELAARD@ACTA.NL
Academic Centre for Dentistry Amsterdam (ACTA), University of Amsterdam and VU University, Amsterdam, The Netherlands

We propose feature selection algorithm based on online co-regularization approach. We demonstrate that by sequentially co-regularizing prediction functions constructed for each view we are able to efficiciently identify landmark features important for the learning process. Furthermore, we evaluate the efficiency and performance of the proposed algorithm on the Human Microbiome Project dataset (Human Microbiome Project 16S rRNA 454 Clinical Production Phase I). In particular we address the task of finding landmark (biomarker) species that are highly predictive of the abundance of Porphyromonas in the oral cavity. In our empirical evaluation the proposed method notably outperforms several feature selection techniques as well as leads to significant computational benefits when training the model.

Multiagent Control as a Graphical Model Inference Problem

Hilbert J. Kappen B.KAPPEN@SCIENCE.RU.NL
Vicenç Gómez V.GOMEZ@SCIENCE.RU.NL
Donders Institute for Brain Cognition and Behaviour
Radboud University Nijmegen 6525 EZ Nijmegen, The Netherlands

Manfred Opper OPPERM@CS.TU-BERLIN.DE
Department of Computer Science
D-10587 Berlin, TU Berlin, Germany

KL-control problems are a certain class of non-linear stochastic optimal control problems for which the optimal control cost C is a Kullback-Leibler divergence between the optimal control law p and the uncontrolled process q plus a state dependent expected cost of future states $\langle R\rangle_p$ (Kappen et al., 2012; Todorov, 2007).

$$C = \mathrm{KL}(p\|q) + \langle R\rangle_p.$$

In this work, we show that this class of problems corresponds to a probabilistic inference problem defined on a factor graph, where variable nodes denote the states of the system at different times and factor nodes encode either the uncontrolled process or the state costs. The optimal control is given by a marginal distribution that can be computed using standard methods such as the junction tree or belief propagation (BP).

We consider the following game defined on a grid where M agents (hunters) can move to adjacent locations for T time steps. The grid also contains hares and stags at fixed locations. Each hunter can choose between hunting a hare on his own, resulting in a small reward R_h, or hunting a stag, resulting in a larger reward $R_s \gg R_h$, but requiring cooperation of two hunters.

To define the factor graph associated to this problem, let x_i^t (variables) denote the position of hunter i at time t on the grid. Also, let s_j and h_k denote the positions of the jth stag and the kth hare respectively. The state dependent reward factor can be written as: $\psi_R(x^t) = \exp(-1/\lambda R(x^t))$,

$$R(x^t) = R_h \sum_{k=1}^{H}\sum_{i=1}^{M} \delta_{x_i^t,h_k} + R_s \sum_{j=1}^{S} \mathcal{I}\{(\sum_{i=1}^{M} x_i^t = s_j) > 1\}.$$

The uncontrolled dynamics factorizes among the agents $\psi_q(x^t|x^{t-1}) = \prod_i \psi_q(x_i^t|x_i^{t-1})$ and is defined as a random walk, allowing an agent to stay or to move to an adjacent position with equal probability.

We "clamp" x_i^0 to a given initial configuration and estimate the marginals (optimal controls) $p(x^{1:T}|x^0)$.

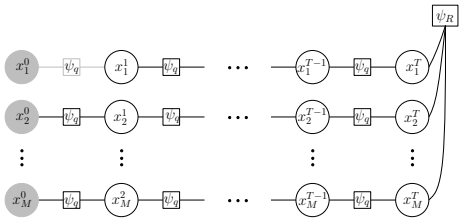


Figure 1. Factor graph representation for the KL-stag-hunt problem. Approximate optimal control can be obtained through the BP factor beliefs between two time slices.

Computing them exactly is intractable, since the state space scales as N^M. BP is an alternative approximate algorithm with polynomial complexity.

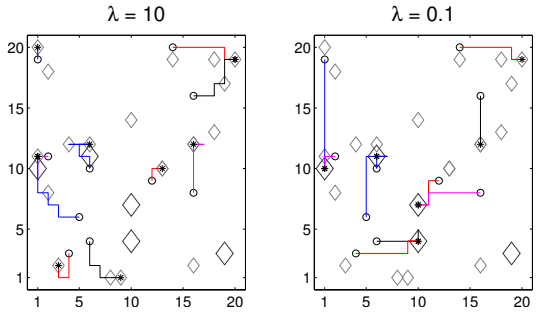


Figure 2. Examples of solutions using BP for 10 agents for different values of λ. (Left) **Risk** dominant optimal control: all hunters go for a hare. (Right) **Payoff** dominant optimal control: hunters cooperate to capture the stags. Small and big diamonds denote hares and stags respectively. Circles denote initial positions.

References

Kappen, H. J., Gómez, V., & Opper, M. (2012). Optimal control as a graphical model inference problem. *Machine Learning*, *87*, 159–182.

Todorov, E. (2007). Linearly-solvable Markov decision problems. In *NIPS 19*, 1369–1376. MIT Press.

Detecting Excessive Claim Behavior in Medical Insurance Claims

Rob M. Konijn KONIJN@LIACS.NL, R.M.KONIJN@VU.NL
Arno Knobbe KNOBBE@LIACS.NL
LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
VU University Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands

Keywords: subgroup discovery, outlier detection, fraud detection

Health care spending is a heavy financial burden that many nations have to face. Rising health care costs are not only caused by aging populations and new medical technologies, but also by unnecessary services, inflated prices, or even fraud. In our research we are interested in finding patterns that correspond to the latter three causes.

When a patient visits a medical practitioner (a dentist, pharmacy, GP, hospital etc.), the practitioner charges an amount of money corresponding to the treatment the patient received. Because the patient generally does not know exactly what service is charged, there is a risk of erroneous claim behavior or even fraud, since the practitioner is the only one who knows the treatment actually performed and charged. Our data, made available by Achmea (the largest Dutch health insurance company), describes claims resulting from treatments provided by several classes of medical practitioners, including dentists, general practitioners (GPs) and pharmacies.

1. Subgroup Discovery

The problem of identifying interesting patterns in claim behavior is essentially an unsupervised learning problem. We have no claims that are labeled as fraudulent beforehand. The data we consider describes patients. A single record summarizes the care a patient received during a certain period (usually one year). The approach we take, is to single out a practitioner and compare its claim behavior against all other practitioners. We assume there is a single practitioner under investigation (the target practitioner). There will be a single target column t with domain $\{0, 1\}$ (or $\{true, false\}$), which identifies whether or not each patient visited this practitioner over a given period of time. Describing differences between target and non-target examples is called Subgroup Discovery (SD).

In (Konijn et al., 2013a) we describe how *local subgroups* of patients can be found. For a local subgroup, patients in the subgroup are much more present at the target practitioner (the target is true more often), while 'similar' patients outside the subgroup are much less frequently present. An example result is found in the case of dentistry, where the subgroup $\{$*Consult, X-Ray picture*$\}$ is much more frequently true than the (roughly similar) subgroup $\{$*Consult*$\}$, meaning that the dentist under investigation charges an X-Ray picture next to a consult more often than other dentists.

Additional to counts of patients, also the costs spent on the (treatments of) patients are important. Subgroups with more money involved, are more interesting. We can view the data as having two target columns: one binary target and one continuous target describing costs. In (Konijn et al., 2013b) we investigate possible quality measures that take into account both the binary target as well as the costs-target. We aim to produce interpretable valuation of subgroups, such that data analysts can directly value the findings, and relate these to monetary gains or losses.

Current research is about including prior knowledge in the SD process. In our application, practitioners that mainly treat old patients will have a different claim distribution than practitioners that mainly treat young patients. To still be able to compare practitioners, we will incorporate this knowledge in the SD process.

References

Konijn, R. M., Duivesteijn, W., Kowalczyk, W., & Knobbe, A. (2013a). Discovering local subgroups, with an application to fraud detection. *PAKDD 2013*.

Konijn, R. M., Duivesteijn, W., Meeng, M., & Knobbe, A. (2013b). Cost-based quality measures in subgroup discovery. *New Frontiers in Applied Data Mining - PAKDD 2013 International Workshops*.

Improving Cross-Validation Classifier Selection Accuracy through Meta-learning

Jesse H. Krijthe J.H.KRIJTHE@LUMC.NL
Leiden University Medical Center, Einthovenweg 20, 2333 ZC Leiden, The Netherlands

Tin Kam Ho TKH@RESEARCH.BELL-LABS.COM
Bell Laboratories, Alcatel-Lucent, 600 Mountain Ave., Murray Hill, New Jersey, 07974-0636, USA

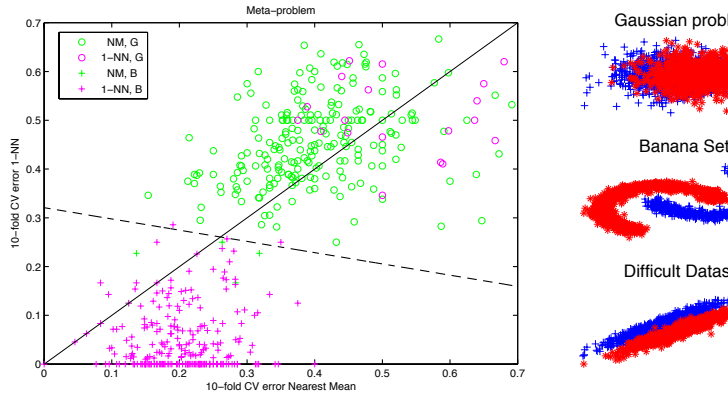
Marco Loog M.LOOG@TUDELFT.NL
Delft University of Technology, Mekelweg 4, Mekelweg 4, 2628 CD Delft, The Netherlands

Keywords: classifier selection, meta-learning, cross-validation

Given the large amount of classication algorithms available, choosing an algorithm for a given dataset is a non-trivial problem. In practice, a cross-validation procedure is often employed to estimate the true errors of a set of classifiers and the classifier with the lowest error estimate is used. However, for small sample sizes, cross-validation error estimates have been shown to become unreliable (Braga-Neto & Dougherty, 2004). Krijthe et. al. (2012) explore whether one can improve classifier selection using techniques from the field of meta-learning. This contribution recapitulates the main finding.

Meta-learning assumes a collection of datasets is given. Selecting a classifier can then be seen as a classification problem on a *meta* level where datasets are the meta-objects and the meta-features can be any measure derived from a dataset. The meta-classes are the classifiers that have the lowest true error on each dataset. One could consider as a special case of meta-features the cross-validation errors of all classifiers under consideration.

As an illustration, the figure shows the meta-problem of a simulated collection of datasets consisting of two base problems. The goal is to choose which of two classifiers would give a lower generalization error. Regular cross-validation selection corresponds to the diagonal boundary in this space. It is clear that the decision boundary of a trained meta-classifier, the dotted line, is markedly different. In fact, when using this meta-classifier the error in selecting the best classifier drops from 0.16 to 0.06. Additionally, adding other meta-features, such as the variance of the cross-validation errors, further improves the classifier selection.



These results corroborate the interesting observation that classifier selection by meta-learning techniques can outperform the de facto standard: cross-validation. Experiments on quasi-real world data suggests these effect may be present in non-simulated data as well. Secondly, the usefulness of adding additional meta-features indicates that not all information relevant in classifier selection may be present in the cross-validation estimates, suggesting improved classifier selection techniques may be possible.

References

Braga-Neto, U., & Dougherty, E. R. (2004). Is cross-validation valid for small-sample microarray classification? *Bioinformatics*, *20*, 374–380.

Krijthe, J. H., Ho, T. K., & Loog, M. (2012). Improving cross-validation based classifier selection using meta-learning. *21st International Conference on Pattern Recognition* (pp. 2873–2876).

MaSh: Machine Learning for Sledgehammer

Daniel Kühlwein DANIEL.KUEHLWEIN@GMAIL.COM
ICIS, Radboud Universiteit Nijmegen, The Netherlands

Jasmin Christian Blanchette BLANCHETTE@IN.TUM.DE
Fakultät für Informatik, Technische Universität München, Germany

Cezary Kaliszyk CEZARY.KALISZYK@UIBK.AC.AT
Institut für Informatik, Universität Innsbruck, Austria

Josef Urban JOSEF.URBAN@GMAIL.COM
ICIS, Radboud Universiteit Nijmegen, The Netherlands

Keywords: Machine Learning, Interactive Theorem Proving, Automatic Theorem Proving

Sledgehammer (Paulson & Blanchette, 2010) is a subsystem of the proof assistant Isabelle/HOL (Nipkow et al., 2002) that discharges interactive goals by harnessing external automatic theorem provers (ATPs). It heuristically selects a number of relevant facts (axioms, definitions, or lemmas) from the thousands available in background libraries and the user's formalization, translates the problem to the external provers' logics, and reconstructs any machine-found proof in Isabelle. The tool is popular with both novices and experts.

Meng and Paulson (Meng & Paulson, 2009) designed a filter, MePo, that iteratively ranks and selects facts similar to the current goal, based on the symbols they contain. Despite its simplicity, and despite advances in prover technology (Hoder & Voronkov, 2011; Schulz, 2011), this filter greatly increases the success rate: Most provers cannot cope with tens of thousands of formulas, and translating so many formulas would also put a heavy burden on Sledgehammer.

MaSh is a learning-based alternative to MePo. It learns from successful proofs, whether human-written or machine-generated. MaSh's heart is a Python program that implements a custom version of a weighted sparse naive Bayes algorithm that is faster than the algorithms used in previous studies (Alama et al., 2011). The program maintains a persistent state and supports incremental, nonmonotonic updates. Although distributed with Isabelle, it is fully independent and could be used by other proof assistants or applications with similar requirements.

The full paper is submitted to ITP 2013, the fourth conference on interactive theorem proving.

Acknowledgments

Daniel Kühlwein is supported by the Nederlandse organisatie voor Wetenschappelijk Onderzoek (NWO) project Learning2Reason. Jasmin Blanchette is supported by the Deutsche Forschungsgemeinschaft (DFG) project Hardening the Hammer (grant Ni 491/14-1).

References

Alama, J., Heskes, T., Kühlwein, D., Tsivtsivadze, E., & Urban, J. (2011). Premise selection for mathematics by corpus analysis and kernel methods. *CoRR*, *abs/1108.3446*. `http://arxiv.org/abs/1108.3446`.

Hoder, K., & Voronkov, A. (2011). Sine qua non for large theory reasoning. *CADE-23* (pp. 299–314). Springer.

Meng, J., & Paulson, L. C. (2009). Lightweight relevance filtering for machine-generated resolution problems. *J. Applied Logic*, *7*, 41–57.

Nipkow, T., Paulson, L. C., & Wenzel, M. (2002). *Isabelle/HOL: A proof assistant for higher-order logic*, vol. 2283 of *LNCS*. Springer.

Paulson, L. C., & Blanchette, J. C. (2010). Three years of experience with Sledgehammer, a practical link between automatic and interactive theorem provers. *IWIL-2010*.

Schulz, S. (2011). First-order deduction for large knowledge bases. Presentation at Deduction at Scale 2011 Seminar, Ringberg Castle.

Learning by Marginalizing Corrupted Features

Laurens van der Maaten LVDMAATEN@GMAIL.COM
Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands

Minmin Chen MC15@CEC.WUSTL.EDU
Stephen Tyree SWTYREE@WUSTL.EDU
Kilian Weinberger KILIAN@WUSTL.EDU
Washington University in St. Louis, St. Louis, MO 63130, USA

Keywords: Regularization, supervised learning, data corruption.

Overfitting is a key problem in machine learning. It is generally combatted using *regularization* or *Bayesian* techniques that employ priors favoring "simple" models over "complex" models. We propose a new approach to counter overfitting, called *marginalized corrupted features* (van der Maaten et al., 2013, MCF). Instead of perturbing models, which can be counter-intuitive, MCF regularizes by *perturbing the data*. We may know that certain *corruptions* of data instances do not affect their label. For example, deleting a few words in a document rarely changes its topic. MCF uses this knowledge to generate additional data that looks like real data: it corrupts the *finite* training set with a corrupting distribution to construct an *infinite* corrupted training set on which the model is trained.

The corrupting distribution, which specifies how training observations $\mathbf{x}$ are transformed into corrupted versions $\tilde{\mathbf{x}}$, is assumed to factorize over dimensions d:

$$p(\tilde{\mathbf{x}}|\mathbf{x}) = \prod_{d=1}^{D} P_E(\tilde{x}_d|x_d;\eta_d).$$

Herein, η_d represents hyperparameters of the corrupting distribution. Corrupting distributions of interest, P_E, include: (1) "blankout" noise in which features are randomly set to zero, (2) Gaussian noise, and (3) Poisson noise in which features are used as rates.

Assume we have a training set $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$ and a loss function $L(\mathbf{x}, y; \Theta)$, with model parameters Θ. A simple approach to use the corrupting distribution is by corrupting each training sample M times, and training on the resulting NM corrupted instances. Such an approach is effective (Vincent et al., 2008), but it lacks elegance and is computationally expensive. MCF addresses these issues by considering the limiting case $M \to \infty$, in which we obtain the *expected loss* under the corrupting distribution:

$$\mathcal{L}(\mathcal{D};\Theta) = \sum_{n=1}^{N} \mathbb{E}[L(\tilde{\mathbf{x}}_n, y_n; \Theta)]_{p(\tilde{\mathbf{x}}_n|\mathbf{x}_n)}.$$

For linear models, this expectation can be computed analytically for many loss functions and corrupting distributions. For a linear model with weights $\mathbf{w}$, the expected value of the *quadratic loss* under the corrupting distribution $p(\tilde{\mathbf{x}}|\mathbf{x})$ is (Chen et al., 2012):

$$\begin{aligned}\mathcal{L}(\mathcal{D};\mathbf{w}) &= \sum_{n=1}^{N} \mathbb{E}\left[\left(\mathbf{w}^{\mathrm{T}}\tilde{\mathbf{x}}_n - y_n\right)^2\right]_{p(\tilde{\mathbf{x}}_n|\mathbf{x}_n)} \\ &= \mathbf{w}^{\mathrm{T}}\mathbf{H}\mathbf{w} - 2\left(\sum_{n=1}^{N} y_n \mathbb{E}\left[\tilde{\mathbf{x}}_n\right]\right)^{\mathrm{T}}\mathbf{w} + N,\end{aligned}$$

where the hat matrix $\mathbf{H} = \sum_{n=1}^{N} \mathbb{E}[\tilde{\mathbf{x}}_n]\mathbb{E}[\tilde{\mathbf{x}}_n]^{\mathrm{T}} + V[\tilde{\mathbf{x}}_n]$, and $V[x]$ is the variance of x. Hence, to minimize the expected quadratic loss, we only need to compute the mean and variance of the corrupting distribution. This is efficient for a wide range of corruption models.

For *logistic loss*, we derive a closed-form upper bound:

$$\begin{aligned}\mathcal{L}(\mathcal{D};\mathbf{w}) &= \sum_{n=1}^{N} \mathbb{E}\left[\log\left(1 + \exp\left(-y_n \mathbf{w}^{\mathrm{T}}\tilde{\mathbf{x}}_n\right)\right)\right]_{p(\tilde{\mathbf{x}}_n|\mathbf{x}_n)} \\ &\leq \sum_{n=1}^{N} \log\left(1 + \prod_{d=1}^{D} \mathbb{E}\left[\exp\left(-y_n w_d \tilde{x}_{nd}\right)\right]_{p(\tilde{x}_{nd}|x_{nd})}\right).\end{aligned}$$

Herein, we recognize a product of moment-generating functions that can be computed efficiently for corrupting distributions in the natural exponential family.

We show the merits of learning with MCF for various models and corrupting distributions. In particular, MCF achieves substantial performance improvements in document classification and domain adaptation.

References

Chen, M., Xu, Z., Weinberger, K., & Sha, F. (2012). Marginalized denoising autoencoders for domain adaptation. *Proceedings of the International Conference on Machine Learning* (pp. 767–774).

van der Maaten, L., Chen, M., Tyree, S., & Weinberger, K. (2013). Learning by marginalizing corrupted features. *Proceedings of the International Conference on Machine Learning, JMLR W&CP*, *28*, 410–418.

Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. (2008). Extracting and composing robust features with denoising autoencoders. *Proceedings of the International Conference on Machine Learning* (pp. 1096–1103).

Traffic Events Identification with a Sensor Network on a Dutch Highway Bridge

Shengfa Miao MIAO@LIACS.NL
Leiden University, The Netherlands
Lanzhou University, China

Arno Knobbe KNOBBE@LIACS.NL
Leiden University, The Netherlands

Keywords: supervised classification, baseline detection, traffic event identification

In the field of Intelligent Transportation Systems (ITS), different equipment is employed to collect data, including video cameras, sensors, loop-detectors, mobile devices and GPS-enabled vehicles. A number of methods are developed to extract traffic events from the collected data, such as vision-based methods (Yoneyama et al., 2004) and time series models (Vespier et al., 2011). Due to environmental factors, such as shadow, lighting, the vision-based methods face the challenge of maintaining detection accuracy (Yoneyama et al., 2004). The time series models can also be problematic on certain types of time series (Vespier et al., 2011).

In this work, we present a supervised method to extract traffic events from the datasets collected with a sensor network installed on a highway bridge. Part of the work was recently published in the international IABSE conference (Miao et al., 2013). The sensor network is is composed of 145 sensors, installed on three cross-sections of one bridge span. We choose one strain sensor on each side to catch traffic events on the bridge.

Traffic events are represented as peaks in the strain signal. In practice, we cannot simply extract these peaks from raw strain signals, because the strain sensors are sensitive to environmental factors. We remove baseline drifts, caused by temperature or traffic jams, with the improved *first derivative method* (Wolfgang et al., 1991), and extract a number of peaks from the preprocessed strain signals. Each peak can be featured as amplitude, duration, area and label. The label indicates the peak type, which is obtained by referring to video streams collected with a camera. According to video streams, the peaks are divided into 5 groups: noise, big vehicle (of two directions), small vehicle (of two directions).

We choose a dataset with a length of 1 hour at night as training dataset. Based the extracted peak features from the training dataset, we create a decision tree using the *C4.5* algorithm in Weka (Hall et al., 2009). We test the obtained model on a testing dataset, which is also collected at night. We succeed in classifying 99.55% of the peaks in the testing dataset. At night, the traffic is not heavy and there are less overlap peaks in the collected signals. In the future, we will work on traffic event identification methods for signals collected during the rush hour.

References

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explorations*, *11*, 10–18.

Miao, S., Knobbe, A., Koenders, E., & Bosma, C. (2013). Analysis of traffic effects on a dutch highway bridge. *Proceedings of IABSE*.

Vespier, U., Knobbe, A., Vanschoren, J., Miao, S., Koopman, A., Obladen, B., & Bosma, C. (2011). Traffic Events Modeling for Structural Health Monitoring. *Proceedings of Intelligent Data Analysis*.

Wolfgang, D., Christian H, R., & Markus, N. (1991). Fast and precise automatic baseline correction of one- and two-dimensional nmr spectra. *Journal of Magnetic Resonance*, *91*, 1–11.

Yoneyama, A., Yeh, C. H., & Kuo, C. C. J. (2004). Robust traffic event extraction via content understanding for highway surveillance system. *2004 IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1679–1682).

AIC for Conditional Model Selection

Thijs van Ommen THIJS.VAN.OMMEN@CWI.NL
Centrum Wiskunde & Informatica, Science Park 123, 1098 XG Amsterdam, The Netherlands

Keywords: model selection, prediction, supervised learning, covariate shift, AIC

In supervised learning applications, AIC and many other popular model selection methods are biased because they implicitly assume that the inputs (covariates) X in the training set take the same values as the inputs X' in the test set. Based on a novel, unbiased expression for KL divergence, we propose FAIC, a *focused* version of AIC that takes the value of X' on the test set into account. Our experiments suggest that if X' substantially differs from X, then FAIC predictively outperforms AIC, BIC and several other methods including Bayesian model averaging.

We introduce FAIC as an adaptation of AIC to supervised learning problems. The aim of AIC (Akaike, 1973) and many other model selection methods is to use the data to find the model g which minimizes

$$-2\,\mathrm{E}_{\mathbf{U}}\,\mathrm{E}_{\mathbf{V}} \log g(\mathbf{V} \mid \hat{\theta}(\mathbf{U})), \tag{1}$$

where $\hat{\theta}$ represents the maximum likelihood estimator in that model, and both random variables are independent samples of n data points each, both following the true distribution of the data. This quantity can be seen as representing that we first estimate the model's parameters using a random sample $\mathbf{U}$, then judge the quality of this estimate by looking at its performance on an independent, identically distributed sample $\mathbf{V}$.

In supervised learning problems such as regression and classification, the data points consist of two parts $u_i = (x_i, y_i)$, and the models are sets of distributions on the *output variable* $\mathbf{y}$ conditional on the *input variable* x (which may or may not be random). We call these *conditional* models. Then (1) can be adapted in two ways: as the extra-sample error

$$-2\,\mathrm{E}_{\mathbf{Y}|X}\,\mathrm{E}_{\mathbf{Y}'|X'} \log g(\mathbf{Y}' \mid X', \hat{\theta}(X, \mathbf{Y})), \tag{2}$$

and, replacing both X and X' by a single variable X, as the in-sample error

$$-2\,\mathrm{E}_{\mathbf{Y}|X}\,\mathrm{E}_{\mathbf{Y}'|X} \log g(\mathbf{Y}' \mid X, \hat{\theta}(X, \mathbf{Y})). \tag{3}$$

The standard expression behind AIC (1) makes no reference to X or X', so that all known versions of AIC end up estimating the in-sample error. However, the extra-sample error (2) is more appropriate as a measure of the expected performance on new data.

To get an estimator for (2), we do not make any assumptions about the processes generating X and X' (so we can deal with covariate shift and with nonrandom inputs) but treat these values as given. A derivation similar to AIC's leads to a penalty term of $k+\kappa_{X'}$ in place of AIC's $2k$; in the case of linear regression,

$$\kappa_{X'} = \frac{n}{n'} \operatorname{tr}\left[{X'}^{\top} X' (X^{\top} X)^{-1}\right],$$

where X, X' represent design matrices and n, n' their respective numbers of data points. Similarly, a small sample corrected version analogous to AIC_C (Hurvich & Tsai, 1989) can be derived and has penalty

$$k + \kappa_{X'} + \frac{(k+\kappa_{X'})(k+1)}{n-k-1}.$$

If our goal is prediction, then X corresponds to the training data, and X' may be replaced by a single point x for which we need to predict the corresponding $\mathbf{y}$. We name this method *Focused AIC*. Note that FAIC may select different models for different values of x. Alternatively, X' may be chosen using (an estimate of) all test inputs if a single choice of model is desired.

Acknowledgments

I thank Peter Grünwald and Steven de Rooij for their valuable comments and encouragement.

References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. *2nd International Symposium on Information Theory* (pp. 267–281).

Hurvich, C. M., & Tsai, C.-L. (1989). Regression and time series model selection in small samples. *Biometrika*, *76*, 297–307.

OpenML: An Open Science Platform for Machine Learning

Jan N. van Rijn JVRIJN@LIACS.NL
Joaquin Vanschoren JOAQUIN@LIACS.NL
Leiden Institute of Advanced Computer Science, Leiden University, P.O. Box 9512, 2300 RA, Leiden, Netherlands

Keywords: Machine Learning, Databases, Meta-learning

Many machine learning studies have been conducted over the past few decades from which much knowledge has been obtained. However, due to space restrictions imposed on publications, these are only published in a highly summarized form (Vanschoren et al., 2012). It is scientifically important that the details of experiments are freely available to anyone for verifiability, reproducibility and generalizability. Therefore, we introduce a novel open science platform for machine learning research, called OpenML[1]. OpenML is a website where researchers can share all their datasets, algorithms and experiments, search for the results of others, and compare directly with the state of the art through controlled experimentation. Beyond the descriptions of algorithms in papers, OpenML allows researchers to share detailed experiments that are comparable with the results of other algorithms. Moreover, OpenML links all experimental results, and all meta-data of algorithms and datasets, for easy future analysis.

Users can define *tasks* which are well-described problems to be solved by a machine learning algorithm or workflow. A typical task would be: *Predict (target) attribute X of dataset Y with maximized predictive accuracy*. Other users are challenged to build algorithms that solve these tasks. The creation of tasks happens on the fly. Whenever a user is searching for tasks on which his algorithm can be run, the system automatically returns all tasks that are potentially of interest to the user. There exists excellent tools that facilitate controlled algorithm evaluation, such as MLComp[2] and Kaggle[3]. OpenML differs from these on key aspects: It is intended for sharing experiments and comparing research results, all information requisite for reproducing the experiments is openly available and the results are stored in a public, queriable database.

An attempt to solve a task is called a *run*. The server provides the input data and stores the output data for every algorithm. The algorithm is executed on the PC of the user. For some tasks, e.g., predictive tasks, it offers more structured input and output. For instance, a *supervised classification* task provides the folds with which a classifier can be trained and expects predictions for all input instances. The server evaluates the predictions and stores the scores for evaluation metrics. Also more general tasks can be defined, e.g., parameter optimization, feature selection and clustering.

We have developed a web API, which facilitates finding and downloading tasks and datasets and uploading implementations and results. This API will be integrated in various machine learning tools, like Weka, R, RapidMiner and KNIME. Given a task, the tools can automatically download all associated input data. Once executed, the result can be uploaded with just one click. For example, in the case of supervised classification tasks, the input consists of a dataset and the folds, and the result is a file containing the predictions.

For each algorithm in the database, an overview page will be generated containing data about all tasks on which this algorithm was run. This provides information about the performance of the algorithm over a potentially wide range of datasets, with various parameter settings. For each dataset a similar overview page is created, containing a ranking of algorithms that were run on tasks with the dataset as input.

[1] `http://www.openml.org/`, beta version
[2] `http://www.mlcomp.org/`
[3] `http://www.kaggle.com/`

Acknowledgments

This work is supported by grant 600.065.120.12N150 from the Dutch Fund for Scientific Research (NWO).

References

Vanschoren, J., Blockeel, H., Pfahringer, B., & Holmes, G. (2012). Experiment databases. A new way to share, organize and learn from experiments. *Machine Learning*, *87*, 127–158.

Multi-label text classification using parsimonious language models

Sicco N.A. van Sas SICCO@DDO.NL
University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands

Maarten Marx MAARTENMARX@UVA.NL
ILPS, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands

Keywords: text classification, information retrieval, parsimonious language models

More and more text documents are available digitally and a need exists to categorize them. Manual indexing is laborious and requires experts, thus there is potential for (semi-)automatic classification of these documents using a controlled vocabulary of concepts. We compare two methods which train classifiers for these concepts. The first method, called JEX, is based the vector space model and was developed by the European Commissions Joint Research Centre (Pouliquen et al., 2003; Steinberger et al., 2012). The second approach, described in this paper, is based on parsimonious language models (PLMs) (Hiemstra et al., 2004) and uses no language-dependent resources. Its parameters are easier to optimize and it outperforms the first approach.

JEX' method is empirically constructed using more than 1500 experiments in which different combinations of formulas and parameters were evaluated. The result is a combination of log-likelihood, which is used to find relevant terms, and a variation of inverse document frequency to calculate the term-weights. Significant improvements are obtained when large (multi-word) stop lists are used and its 9 parameters are fine-tuned.

The PLM method estimates a concept classifier by comparing the language used in documents labeled with that concept to the language used in the whole corpus. Terms which are well enough *explained* in the whole corpus are given a probability of zero, thereby reducing the size of the model and acting as an automatic stop list.

Both methods were trained and compared on two political datasets, Acquis and Dutch parliamentary questions (PQ). We used 19 languages from the Acquis dataset with each between 20.000-42.000 European legislation documents labeled with concepts from the EuroVoc taxonomy, which consist of 6797 hierarchically structured concepts. The PQ dataset contains nearly 40.000 documents labeled with a smaller taxonomy of 111 concepts. The first chronological 90% of the data is used as train set, the final 10% as test set.

Versions of the PLM system based on unigrams and bigrams where found to perform well, as shown in Table 1. The multi-label classifiers yield a ranked list of concepts for each document and are evaluated using R-precision (Rprec) and mean average precision (MAP). The unigram PLM system significantly outperformed JEX in 11/19 languages on the Acquis dataset (and performed similar in 3 other languages), while the bigram version does this for all 19 languages. Both unigram and bigram systems reach significantly higher scores on the PQ dataset. The PLM system uses only 3 parameters though the results across different datasets and languages were obtained using parameters optimized on a single language.

Table 1. Scores for the Dutch Acquis and PQ datasets. Significance tested with two-tailed paired t-tests $^{\blacktriangle} = p < 0.01$.

	JEX (baseline)		**unigram PLM**		**bigram PLM**	
dataset	**Rprec**	**MAP**	**Rprec**	**MAP**	**Rprec**	**MAP**
Acquis (NL)	0.5527	0.5770	0.5576	0.5762	0.5673$^{\blacktriangle}$	0.5906$^{\blacktriangle}$
PQ	0.4120	0.5491	0.4807$^{\blacktriangle}$	0.6197$^{\blacktriangle}$	0.5175$^{\blacktriangle}$	0.6436$^{\blacktriangle}$

References

Hiemstra, D., Robertson, S., & Zaragoza, H. (2004). Parsimonious language models for information retrieval. *Proc. SIGIR'04* (pp. 39–46).

Pouliquen, B., Steinberger, R., & Ignat, C. (2003). Automatic annotation of multilingual text collections with a conceptual thesaurus. *Proc. EUROLAN'03* (pp. 9–28).

Steinberger, R., Ebrahim, M., & Turchi, M. (2012). JRC Eurovoc Indexer JEX - a freely available multi-label categorisation tool. *Proc. LREC'12*.

Learning Relations: Pitfalls and Applications

Michiel Stock, Willem Waegeman, Bernard De Baets MICHIEL.STOCK@UGENT.BE
KERMIT, Department of Mathematical Modelling, Statistics and Bioinformatics,
Ghent University, Coupure links 653, 9000 Ghent, Belgium

Keywords: learning relations, kernel methods, model evaluation, bioinformatics

Modeling the interactions between different types of agents is of importance in many domains. Social network sites are interested to predict whether two people know each other and e-commerce websites spend huge budgets to be able to recommend the right items to visitors. Also, in biology predicting interactions between different biomolecules is one of most basic steps in a systems biology approach. At a more abstract level, these different settings are identical. There are two sets of complex objects, with some kind of feature description. In addition, a relation matrix with at least some observed elements is available, denoting the interaction or relation between these objects. The problem boils down to constructing a model that is able to fill the missing values of this relation matrix.

Our methods consist of a general framework for solving these types of problems based on a joint feature representation of pairs of objects by means of the Kronecker product pairwise kernel (KPPK). In this most general case the Kronecker is taken between the two object kernels. As such a pairwise kernel matrix is obtained which encodes the covariance between two pairs of objects. This pairwise kernel matrix can subsequently be used in any kernel-based learning algorithm, such as support vector machines or kernel ridge regression. By using kernels it is possible to work with complex structured objects, such as sequences, graphs or trees. Furthermore was shown that the KPPK can be used to model arbitrary relations and can be modified for specifically learning symmetric or reciprocal relations (Waegeman et al., 2012).

Despite the strong theoretical foundations of relational learning, many open questions still remain. Currently, we are interested how to perform testing and cross validation in such settings. For example, one expects the model to make more accurate predictions for new combinations of objects that were included in the training set than for a pair of previously unseen objects. Recently in Nature, this effect was observed in a large-scale experiment (Park & Marcotte, 2012). How models for these different cases should be trained and how their performance is affected is still unclear. Furthermore we are also interested to which degree our models directly use the features of the objects, rather than indirectly exploiting the structure of the relation matrix.

A strong foundation of our framework would allow us to draw links to many other machine learning settings. Per definition, our framework can be used for general collaborative filtering problems, such as information retrieval and recommender systems. For this we have used a KPPK in an efficient ranking setting (Pahikkala et al., 2012). It is also possible to view at our models as multivariate regression or structured output prediction. If one of the types of objects could represent a certain task, our framework could be used for multi-task learning. If the relation matrix is subjected to certain restrictions, it could be used for graph matching. Trying to find a generalized foundation between these different settings is an exciting part of our study.

Acknowledgments

We acknowledge the support of Ghent University (MRP Bioinformatics: from nucleotides to networks).

References

Pahikkala, T., Airola, A., Stock, M., De Baets, B., & Waegeman, W. (2012). Efficient regularized least-squares algorithms for conditional ranking on relational data. *Machine Learning*, *Submitted*.

Park, Y., & Marcotte, E. M. (2012). Flaws in evaluation schemes for pair-input computational predictions. *Nature Methods*, *9*, 1134–1136.

Waegeman, W., Pahikkala, T., Airola, A., Salakoski, T., Stock, M., & De Baets, B. (2012). A kernel-based framework for learning graded relations from data. *IEEE Transactions on Fuzzy Systems*, *99*, 1.

Semi-supervised Multi-view Gaussian Processes for Microbial Growth Prediction

Evgeni Tsivtsivadze EVGENI.TSIVTIVADZE@TNO.NL
Eveline Lommen EVELINE.LOMMEN@TNO.NL
Roy Montijn ROY.MONTIJN@TNO.NL
Jos van der Vossen JOS.VANDERVOSSEN@TNO.NL
MSB Group, The Netherlands Organization for Applied Scientific Research,Zeist, The Netherlands

We propose semi-supervised multi-view Gaussian process (GP) model for microbial growth prediction. Our semi-supervised GP model is formulated using co-regularization approach, namely we construct GPs for different views, such that the training error of each hypothesis on the labeled data is small and, at the same time, the hypotheses give similar predictions for the unlabeled data Our model is naturally suitable for taking into account multiple data representations and learning complex non-linear interactions. We apply proposed model for describing and predicting growth, succession, and proliferation of microbial species in the spoilage process. In our empirical evaluation on the recently collected biological dataset the proposed approach notably outperforms several regression techniques and leads to better understanding of the role of various bacterial species and their in uence on spoilage process.

White-box optimization from historical data

Sicco Verwer S.VERWER@CS.RU.NL
Radboud University Nijmegen, P.O. Box 9010, 6500 GL, Nijmegen, The Netherlands

Qing Chuan Ye, Yingqian Zhang {YE,YQZHANG}@ESE.EUR.NL
Erasmus University Rotterdam, Burg. Oudlaan 50, 3062 PA, Rotterdam, The Netherlands

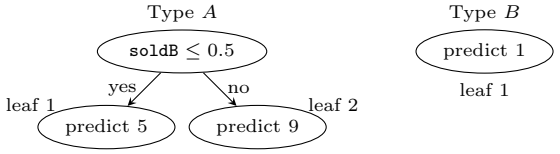
Keywords: optimization, mathematical modeling, machine learning, auction design

It is challenging to construct a mathematical model describing the properties of a system, especially when the structure of the system cannot be fully determined from the hypotheses at hand. In such cases, machine learning techniques can be used to replace (parts of) a mathematical decision model. However, the models produced by machine learning have so far only been used in a black-box fashion, e.g. as fitness functions or parameters. We propose a *white-box* optimization by mapping a learned regression tree model to a mixed integer linear program that can be used for optimization. Consequently, the learned model's properties are visible as constraints to a mathematical problem solver, that can then use sophisticated branching and cutting techniques on these constraints when finding solutions, which are impossible in black-box optimization.

We illustrate our approach using a sequential auction design problem. The objective is to maximize the expected revenued of the auctioneer with multiple bidders (agents) who have complementary preferences over items. We try to find the optimal *ordering of items* to maximize the expected revenue. This problem is proven to be NP-complete. We learn the overall preferences of the group of bidders from historical data by viewing the prediction of the revenue of an auction as a regression problem. We split this problem into the subproblems of predicting the revenue of the auctioned items, and then sum these up to obtain the overall objective function.

We use the following simple example to demonstrate the use of our method. Two agents a_1 and a_2 partake in a sequential auction of items A and B. Their valuations are given by $v_{a_1}(A) = 1, v_{a_1}(B) = 1, v_{a_1}(\{A, B\}) = 10, v_{a_2}(A) = 5$. Two past auctions are known: A was sold first to a_2 and then B to a_1 with a total revenue of 6, whereas in the second auction, B was sold first and then A, both to a_1 with a total revenue of 10. We construct feature values from two auctions, as shown in the table. `sold(.)` represents how many items of type `(.)` have been sold prior to the current item. For each of item types, we learn a regression tree (see the figure).

type	value	soldA	soldB	type	value	soldA	soldB
A	5	0	0	B	1	0	0
B	1	1	0	A	9	0	1



Suppose we are given the set of items $\{A, B, B\}$ for which we need to find an optimal ordering. We can then use the learned trees to formulate the problem of finding an optimal ordering for this set of items as an integer linear program (ILP), which can then be solved by one of many ILP-solvers:

$$
\begin{array}{rlll}
\text{max} & \sum_{1\le i\le 3} 5z_{i,1,A} + 9z_{i,2,A} + 1z_{i,1,B} & & \\
\text{subject to} & & & \\
& x_{i,A} + x_{i,B} & = 1 & \text{for all } 1 \le i \le n \\
& x_{1,A} + x_{2,A} + x_{3,A} & = 1 & \\
& x_{1,B} + x_{2,B} + x_{3,B} & = 2 & \\
& \texttt{sold}_{1,B} = 0 \quad \texttt{sold}_{2,B} = x_{1,B} \quad \texttt{sold}_{3,B} = x_{1,B} + x_{2,B} & & \\
& y_{i,\texttt{sold}_{B>0.5}} & < 1.0 + \frac{\texttt{sold}_{1,B}-0.5}{100} & \text{for all } 1 \le i \le n \\
& y_{i,\texttt{sold}_{B>0.5}} & \ge \frac{\texttt{sold}_{1,B}-0.5}{100} & \text{for all } 1 \le i \le n \\
& y_{i,\texttt{sold}_{B\le 0.5}} & = 1.0 - y_{i,\texttt{sold}_{B>0.5}} & \text{for all } 1 \le i \le n \\
& z_{i,1,A} & \le \frac{x_{i,A}+y_{i,\texttt{sold}_{B\le 0.5}}}{1+0.5} & \text{for all } 1 \le i \le n \\
& z_{i,2,A} & \le \frac{x_{i,A}+y_{i,\texttt{sold}_{B>0.5}}}{1+0.5} & \text{for all } 1 \le i \le n \\
& z_{i,1,B} & \le \frac{x_{i,B}}{0.5} & \text{for all } 1 \le i \le n
\end{array}
$$

Although optimizing the orderings in sequential auctions is a well-known hard problem, our method obtains very high revenues, significantly outperforming the greedy and random methods proposed in the literature. Our constructions are general and can be applied to any settings where regression trees can be learned from data, and their feature values can be computed as linear functions from solutions.

Supermodels: Dynamically Coupled Imperfect Models

Wim Wiegerinck W.WIEGERINCK@SCIENCE.RU.NL
Willem Burgers W.BURGERS@SCIENCE.RU.NL
Donders Institute for Brain Cognition and Behaviour
Radboud University, Nijmegen, The Netherlands

Frank Selten SELTEN@KNMI.NL
Royal Netherlands Meteorological Institute, De Bilt, The Netherlands

In weather and climate prediction studies multi-model ensemble mean predictions are often employed to improve prediction skills (Tebaldi & Knutti, 2007). In the standard multi-model ensemble approach, the models are integrated in time independently and the predicted states are combined a posteriori. Recently an approach has been proposed in which the models exchange information during the simulation (van den Berge et al., 2011). This approach is called the supermodeling approach (SUMO). It assumes M imperfect models labeled by μ, each describing the dynamics of the model state vector $\boldsymbol{x}_\mu$ according to $\dot{x}^i_\mu = f^i_\mu(\boldsymbol{x}_\mu)$, in which i labels the vector components. The individual models μ are combined into one supermodel by prescribing nonnegative connections $C^i_{\mu\nu}$ between the i-th component of model μ and model ν,

$$\dot{x}^i_\mu = f^i_\mu(\boldsymbol{x}_\mu) + \sum_\nu C^i_{\mu\nu}(x^i_\nu - x^i_\mu)\,. \qquad (1)$$

The connections are to be optimized using data, for instance by minimizing short term prediction error of the connected ensemble mean on the training data. Its potential has been demonstrated in the context of 3-D chaotic dynamical systems. With optimized connections, the models synchronize on a common solution that is closer to the true system than any of the individual model solutions.

In (Wiegerinck et al., 2013a), we have shown that with large connections, the SUMO follows approximately the weighted averaged trajectory

$$\dot{x}^i = \sum_\mu w^i_\mu f^i_\mu(\boldsymbol{x})\,. \qquad (2)$$

where the weights $\{w^i_\mu\}$ can be derived from eigenvectors of the connection matrices. Also, with (2), we could understand local minimima in the connection space and results due to parameter perturbations reported in (van den Berge et al., 2011).

In (Wiegerinck et al., 2013b), we defined the supermodel directly according to (2), which we called weighted SUMO (opposed to connected SUMO in (1)). While connected SUMO needs nonlinear optimization, weighted SUMO can be optimized using linear optimization methods, making the method scalable for models of higher dimensions. We demonstrated the method in the context of a two-level, hemispheric, quasi-geostrophic spectral model on the sphere, triangularly truncated at wave number five, with 30 degrees of freedom (Houtekamer, 1991).

Acknowledgments This work has been supported by FP7 FET Open Grant # 266722 (SUMO project). The work reported here has been published in (Wiegerinck et al., 2013a; Wiegerinck et al., 2013b).

References

Houtekamer, P. (1991). Variation of the predictability in a low-order spectral model of the atmospheric circulation. *Tellus A*, *43*, 177–190.

Tebaldi, C., & Knutti, R. (2007). The use of the multi-model ensemble in probabilistic climate projections. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *365*, 2053.

van den Berge, L. A., Selten, F. M., Wiegerinck, W., & Duane, G. S. (2011). A multi-model ensemble method that combines imperfect models through learning. *Earth System Dynamics*, *2*, 161–177.

Wiegerinck, W., Burgers, W., & Selten, F. (2013a). On the limit of large couplings and weighted averaged dynamics. In L. Kocarev (Ed.), *Consensus and synchronization in complex networks*, 257 – 275. Springer.

Wiegerinck, W., Mirchev, M., Burgers, W., & Selten, F. (2013b). Supermodeling dynamics and learning mechanisms. In L. Kocarev (Ed.), *Consensus and synchronization in complex networks*, 227 – 255. Springer.

Empirical Training For Conditional Random Fields

Zhemin Zhu z.zhu@utwente.nl
Djoerd Hiemstra D.Hiemstra@utwente.nl
Peter Apers P.M.G.Apers@utwente.nl
Andreas Wombacher a.wombacher@utwente.nl
CTIT Database Group, Drienerlolaan 5, 7500AE, Enschede, The Netherlands

Conditional Random Fields (CRFs) are undirected graphical models which have been widely applied for sequence labelling, e.g. part-of-speech tagging. Training CRFs (Lafferty et al., 2001) can be very expensive for large-scale applications (Sutton & McCallum, 2009). The standard training **(SD)** of CRFs needs to calculate the partition function $Z_{sd}(X)$ which is a global summation over the whole graph. Piecewise training (**PW**) (Sutton & McCallum, 2009) speeds up the training process by approximating the partition function with an upper bound. But piecewise training is still not scalable to the variable cardinality. Another option for sequence labelling is directed models such as Maximum Entropy Markov Models (MEMMs) (McCallum et al., 2000) which can be trained efficiently. But they suffer from the label bias problem (Lafferty et al., 2001) which may lead to low accuracy.

In this paper (Zhu et al., 2013), we present a practically scalable training method for CRFs called *Empirical Training* **(EP)**. We show that the standard training with unregularized log likelihood can have many maximum likelihood estimations (MLEs). Empirical training has a unique closed form MLE which can be calculated from the empirical distribution very fast. The MLE of the empirical training is also one MLE of the standard training. So empirical training can be competitive in precision to the standard training and piecewise training. And also we show that empirical training is unaffected by the label bias problem even it is a local normalized model. Experiments on two real-world NLP datasets also show that empirical training reduces the training time from weeks to seconds, and obtains competitive results to the standard and piecewise training on linear-chain CRFs, especially when training data are insufficient.

Experiment 1. Brown Corpus is used for the Part-of-Speech (POS) tagging experiment. The size of the tag space is 252. There are 32,623 sentences are used for training and 1,000 sentences are used for testing.

Table 1: Part-of-Speech Tagging Accuracy

Metric	**EP**	SD	PW	PWPL
Accuracy	95.6	95.4	82.9	82.4
Time (s)	3.9	4,571,807	3,791,648	261,021

The method may also suffer from some potential drawbacks. When using large feature vectors the empirical probabilities may become sparse, generalisation from the training data to the test data may be a problem. Also in the experiment we did not try global features. So there is no evidence to show this method works well with global features. Nevertheless, this method is very fast and could be very useful for practitioners who apply CRFs to large scale data sets.

Acknowledgments

This work has been supported by the Dutch national program COMMIT/.

References

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML* (pp. 282–289).

McCallum, A., Freitag, D., & Pereira, F. C. N. (2000). Maximum entropy markov models for information extraction and segmentation. *ICML* (pp. 591–598).

Sutton, C. A., & McCallum, A. (2009). Piecewise training for structured prediction. *Machine Learning*, *77*, 165–194.

Zhu, Z., Hiemstra, D., Apers, P. M. G., & Wombacher, A. (2013). *Closed form maximum likelihood estimator of conditional random fields* (Technical Report TR-CTIT-13-03). CTIT, University of Twente.