Netflix Challenge HSR

Netflix Challenge @ HSR

Resultate des Kurses „Challenge Projekte“ HSR 2009

2009 Herbst Thema „Data Mining in grossen Data Sets: Der Netflix Preis“

Topic Fall Semester 2009: „Data Mining in large Dataset: The Netfix Prize“

Hintergrund:

An der HSR wird seit 2009 im Frühjahr und Herbst das Modul „Challenge Projekte“ angeboten. Im Rahmen dieses Moduls können Studenten der HSR an Informatik Wettbewerben teilnehmen. Dies sind keine Individualaufgaben sondern alle Studenten des Moduls nehmen am gleichen Wettbewerb teil und werden vom Dozenten und eingeladenen Gastrednern in Ihrer Arbeit unterstützt (Modul-Details http://unterricht.hsr.ch/staticWeb/allModules/18057_M_ChallP.html). Im Herbst 2009 war das Thema des Challenge Projekte Moduls der Netflix Preis.

Netflix ist einer der führeneden Vermieter von DVDs in den USA. Kunden lösen ein Abbonement für eine bestimmte Anzahl von DVDs. Sie erhalten dann die DVDs welche sie auf Ihrer Internet-Wunschliste aufgeführt haben per Post und senden diese auch wieder per Post zurück. Ein wichtiger Erfolgsfaktor für Netflix ist ihr „Recommender System“. Dies ist eine Beratungskomponente im Netflix Internet Portal welche Kunden Filme vorschlägt die ihnen gefallen könnten. Diese Vorschläge werden berechnet aufgrund von Bewertungen welche Kunden für die Filme abgeben welche sie schon gesehen haben. Netflix hatte realisiert, dass die Qualität der Vorschläge als einen kritischen Erfolgsfaktor darstellt. Daher schrieb Netflix am 2. Oktober 2006 den „Netflix Prize“ aus (http://www.netflixprize.com/). Dieser Preis war mit einer Million US$ dotiert für das Team welche es schaffte den aktuellen Algorithmus des Netflix Recommender Systems um 10% zu verbessern. Neben der hohen Preissumme war die Besonderheit des Preises, dass die Teilnehme weltweit offen war, ein sehr grosses Datenset verfügbar gemacht wurde (100 Millionen Ratings von 480‘000 anonymisierten Nutzern zu 18‘000 Filmen), und dass ein automatischer Service zur Lösungsevaluation zur Verfügung gestellt wurde welcher die Daten auf einem öffentlich einsehbaren „Leaderboard“ veröffentlichte. Trotz dieser optimalen Bedingungen wurde das Ziel von 10% Verbesserung bis 2009 nicht erreicht. Daher wurden entsprechend dem Preis-Regelement im Oktober 2007 und im Oktober 2008 „Progress Pizes“ vergeben. Der erste Progress Preis (2007) ging an das Team KoreBell (AT&T Labs: Y. Koren, R. Bell & C. Volinsky). Der zweite Progress Preis (2008) ging an den Verbund von zwei Teams (Commendo, Österreich: A. Töscher und M. Jahrer; AT AT&T Labs: Y. Koren, R. Bell & C. Volinsky).

Details zum HSR Modul

Zum Zeitpunkt der Ausschreibung des HSR Challenge Projeke Moduls (Juni 2009) war der Wettbewerb noch immer offen, aber am 26 Juni machte das Commendo & AT&T Team ihre erste Einreichung mit über 10% Verbesserung. Daher wurde der offizielle Wettbewerb, entsprechend den Preisregeln am einen Monat später, am 26 Juli 2009 geschlossen. Am 18 September wurde dann bekannt gegeben, dass das Commendo & AT&T Team mit 10.09% Verbesserung (RMSE of 0.8554) Gewinner des Wettbewerbes sind. Damit musste die 12 Teilnehmer (6 Teams) des HSR Challenge Modul ausser Konkurrenz ihre Aufgaben bewältigen. Hierfür wurde ein HSR interner Evaluations-Service aufgebaut. Den Teams standen besonders ausgerüstete PCs zur Verfügung (Dual Processor, Quad-Core PCs mit 24 GB RAM). Die meisten der Studenten hatten im Semester vorher einen Kurs belegt in dem Sie viele der Grundlegenden Data Mining Algorithmen kennengelernt hatten. Von KNN, <@@@@ bis RBM @@@@> . Jetzt konnten sie anhand der Netflix Challenge ihre erworbenen Kenntnisse vertiefen. In der Einführungsveranstaltung am 18. September stelle Markus Stolze (HSR) nochmals die grundlegenden Typen, Algorithmen und Techniken von Recommender Systemen vor, Eric Cope von IBM Forschungslabor in Rüschlikon gab eine Einführung in SVD und die Arbeit mit den Statistikpaket R, und Josef Joller (HSR) besprach den Hintergrund von fortgeschrittenen Netflix Prize Recommender Techniken wie sie in publizierten ACM und IEEE Artikeln dargestellt wurden. Bis zur „Halbzeitveranstaltung“ am 30. Oktober hatten sich alle Teams in den Anwendungsbereich und die Algorithmen eingearbeitet, es geschafft das grosse Datenset einzulesen und eine erste Submission zu manchen.

Die folgenden Teams und Studenten nahmen Teil:

<@@@@

- Gruppe 6: Simon Keller & Dan Krusi, "Linflix"

@@@@>

Halbzeitveranstaltung

Für die Halbzeitveranstaltung hatten wir Andreas Töscher, einen der Gewinner des Netflix Preises zum Vortrag einladen können. In seinem Vortrag und hinterher in der Diskussion mit den Teams wurden verschiedene Fragen diskutiert welche vorher unklar geblieben waren.

Abschluss und Resultate

In der zweiten Phase des HSR internen Wettbewerbes war die Submission offen bis Mittwoch 16.12. Hier gab es insgesamt <@@@@ (Gruppe6:5) @@@@> offizielle Submissions der Teams. Es hatten sich verschiedene „Camps“ gebildet. Ein Java Camp (2 Teams) ein C++ Camp (2 Teams) und zwei weitere einzel-Teams von denen das eine auf Linux und C++ arbeitete, das andere auf C++.

Von den Teams wurden die existierenden Frameworks evaluiert und die besten adaptiert. Die folgenden Frameworks wurden evaluiert: <@@@@ Netflix Prize Tools, Icefox, Kadri @@@@>. Diese Frameworks wurden wurden wieter entwickelt: <@@@@ Kadri @@@@> Die folgenden Algorithmen wurden eingesetzt: <@@@@ Average, Globals, KNN, UKNN, RBM, SVD @@@@> Dabei ergaben sich die folgenden „Findings“ zu den Algorithmen Auch weitere Lessons Learned („What nobody told us about the Netflix data and algorithms“) wurden gesammelt. Hier eine Auswahl (mit Team Name) <@@@@@@@@@@@@@@@>

Results Simon Keller & Dan Krusi

Environment

- OS: Linux 64bit
- Compiler: g++
- Libraries: LAPACK, Qt 4.5

Software

- Linflix, hosted by SourceForge, based on the work by Benjamin C. Meyer and Saqib Kadri

Algorithms

- Random
- Average
- Globals
- SVD
- SVD with Bias
- SVD++
- KNN
- UKNN
- RBM

Source

- GPLv3
- SVN: svn co https://linflix.svn.sourceforge.net/svnroot/linflix
- Web: http://linflix.sourceforge.net

Setup and Building

- Grab source code from SVN:
    - svn co https://linflix.svn.sourceforge.net/svnroot/linflix
- Install necessary software (compiler, frameworks):
    - sh linflix/scripts/setup.sh
- Build:
    - sh linflix/scripts/build_all.sh
- Prepare data:
    - sh linflix/scripts/preparedata.sh
- Run a model, build a preprocessor, and save results:
    - sh linflix/scripts/run.sh rbm

Usage

```
Usage: linflix [OPTIONS] [ALGORITHMS]


OPTIONS:
  --debug: Enable debug output information
  --no-fulloutput: Disable full RMSE output
  --help: Shows usage information
  --build-preprocessor: Build a pre-processor for blending
                        Automatically selected when blending
```

```
ALGORITHMS:
  --random: Random
  --average: Average
  --globals: Globals
  --svd: SVD
  --svdbias: SVD with bias
  --svdpp: SVD++
  --knn: K-Nearest Neighbor
  --uknn: User K-Nearest Neighbor
  --rbm: Restricted Bolzman Machine


BLENDING:
  --blend: Blending of models
  --blendpartial: Partial blending of models
```

Official Submissions

- 20.10.2009: 0.9236976 (SVD)
- 24.11.2009: 0.9197537 (SVD with Bias)
- 14.12.2009: 0.91486906 (RBM)
- 15.12.2009: 0.90395449 (Blending 3xMF, UKNN)
- 16.12.2009: 0.89276278 (Blending 3xMF, UKNN, RBM)

Strategy

- Extend existing open-source software in the direction of the leading Netflix Prize teams
- Integrate the work of Benjamin C. Meyer and Saqib Kadri
- Implement our own features and solutions
- Provide a framework which is easy to setup, build, and run

Biggest Problems

- Insufficient mathematical understanding
  - It's difficult to freshly dive into recommender systems in such little time
- Testing and Verification
  - Tests are difficult to implement in a data subset environment
  - Many algorithms running with just one pass take several minutes to run – slow development process
- Time Deficiency
  - Our best models can take several weeks to run in full glory

More Information

- Dan Krusi
  - dan@nerves.ch
- Simon Keller
  - s1keller@hsr.ch

- Linflix:
  - http://linflix.sourceforge.net

Resultate Pascal Albrecht & André Nanz

Evaluierte Software:

- Wolfram Mathematica 7.0 / Performance Probleme / Speicherverbrauch
- JNetflix / Unsauberer Code
- Taste / Uneffiziente Implementierung
- Duine / Datenhaltung über Spring&Hibernate
- Icefox netflixrecommenderframework / Effizient, wenig Infrastruktur

Fazit

- Je höher der Abstraktionsgrad, desto langsamer wird gerechnet. (C++ bevorzugen vor Mathematica, Ruby, Perl, Python und so weiter)
- Je speziefischer ein Problem, desto unschöner der Code. (Vorgaben nicht refactoren, sondern am Problem arbeiten)
- Bei fixen Eingabedaten entsprechen die Ausgabedaten "dem Algorithmus" (Resultate persistieren)
- Lange Rechenzeiten benötigen eine Form von Parameterpersistenz. (Zwischenresultate zwischenspeichern, USV und so weiter.)
- Das normale Windows geht in den Stromsparmodus, und rechnet nicht weiter. (Stromspar-Einstellungen überprüfen)

Algorithmen

- Average
- SVD
- SVD++