Sudoku

Schwierigkeitseinstufung und Generator

Projektarbeit IP-5



Studiengang Informatik

Betreuer: Manfred Vogel, Lucas Brönnimann

Auftraggeber: Rätsel Agentur AG

Verfasser: Matthias Keller, Simon Beck

Brugg, 20.01.2017

# Abstract

Notwendig.

Die Zusammenfassung richtet sich an Interessierte, die sich einen raschen Überblick über den Inhalt eines Berichts

verschaffen wollen. Die Zusammenfassung soll dazu die folgenden drei Aspekte beleuchten:

 Problemstellung

 Vorgehen (gemeint ist ein knapper Hinweis auf die Art der Problembearbeitung, Methodik1)

 Hauptergebnisse (sollen ca. zwei Drittel der Zusammenfassung ausmachen)

Statt einer Zusammenfassung oder zusätzlich dazu kann ein Abstract gefordert sein (vor allem bei

wissenschaftlichen Fachartikeln anzutreffen). Ein Abstract umfasst ca. 10 Zeilen mit wenigen searchable keywords

(das sind Stichwörter/Schlagwörter in englischer Sprache), es enthält aber keine Resultate.

Zusätzlich zu einer Zusammenfassung enthält ein Technischer Bericht ein Management oder Executive

Summary. Ein Management Summary richtet sich in der Regel an Mitglieder des Managements, an Personen

also, die Entscheidungen treffen. Es informiert sachlich, kompakt und präzise (eine bis höchsten zwei A4-

Seiten), aber lesefreundlich über:

1 Zum Beispiel: Literaturrecherche, Analyse, Befragung, Konstruktion, Test, Messungen, Experimente etc.

Leitfaden Berichte 160107 6

 Ausgangslage: Es werden Thema, Auftraggeber und Problemstellung skizziert und die daraus

erwachsenden Aufgaben und zu erreichenden Ziele

 Vorgehen: Darlegen der Vorgehensweise, der eingesetzten Methoden zur Problemlösung, so dass

nachvollziehbar wird, wie die Ergebnisse erreicht wurden

 Hauptergebnisse: Sie bilden den zentralen Teil. Die konkreten (auch unerwarteten) Resultate des

Projektes sind konkret zu nennen und einzuordnen. Welche Resultate sind warum positiv, welche

warum negativ?

 Ausblick: Er werden Empfehlungen abgegeben und mögliche nächste Schritte skizziert: In

welcher Weise kann der Auftraggeber die Ergebnisse und Lösungsvorschläge verwenden? Welche

alternativen Entscheide sind möglich? Welche Probleme bleiben? Welche Risiken bestehen? Wie

geht es weiter?

Zusammenfassung der Arbeit auf einer halben bis max. einer ganzen Seite:

1. Was wurde erreicht?

2. Welche weiteren Informationen findet ein Leser/eine Leserin im

Dokument?

Betrachten Sie das Abstract als eigenständigen Werbetext, der auf einer

Webseite steht. Anhand dieses Textes sollen interessierte Personen motiviert

werden, ihren ganzen Text zu lesen.

Das Abstract wird erst am Schluss geschrieben. Es ist aber enorm wichtig für den

ersten Eindruck.

# Inhaltsverzeichnis

[1 Abstract 2](#_Toc472842639)

[2 Inhaltsverzeichnis 4](#_Toc472842640)

[3 Einleitung 6](#_Toc472842641)

[4 Konzepte 7](#_Toc472842642)

[4.1 Anwendungsdomäne/Umfeld 7](#_Toc472842643)

[4.1.1 Herkunft 8](#_Toc472842644)

[4.1.2 Aufbau und Regeln 8](#_Toc472842645)

[4.2 Lösen von Sudokus 9](#_Toc472842646)

[4.2.1 Lösungsmethoden welche Werte setzen 10](#_Toc472842647)

[4.2.2 Lösungsmethoden welche Pencilmarks verringern 10](#_Toc472842648)

[4.3 Schwierigkeitseinstufung 11](#_Toc472842649)

[4.3.1 Modell 12](#_Toc472842650)

[4.4 Generierung 13](#_Toc472842651)

[5 Technische Umsetzung 13](#_Toc472842652)

[5.1 Technologien 13](#_Toc472842653)

[5.2 Abbildung des Sudoku-Spielfelds 14](#_Toc472842654)

[5.3 Datensätze 14](#_Toc472842655)

[5.3.1 Datenaufbereitung für Statistik 15](#_Toc472842656)

[5.3.2 Datenanalyse 15](#_Toc472842657)

[5.4 Lösungsmethoden 17](#_Toc472842658)

[5.5 Schwierigkeitseinstufung 17](#_Toc472842659)

[5.5.1 Matlab 17](#_Toc472842660)

[5.5.2 Neuroph 18](#_Toc472842661)

[5.6 Generierung 18](#_Toc472842662)

[6 Resultate 19](#_Toc472842663)

[6.1 Lösungsmethoden (gewisse kaum gebraucht) 19](#_Toc472842664)

[6.2 Schwierigkeitseinstufung (guter approach lösungsmethoden) 19](#_Toc472842665)

[6.3 Matlab/Neuroph 19](#_Toc472842666)

[6.4 Generator, verteilung der Schwierigkeitsstufen (geschwindigkeit) 21](#_Toc472842667)

[6.5 21](#_Toc472842668)

[6.6 Generierung 23](#_Toc472842669)

[7 Erweiterungen 24](#_Toc472842670)

[7.1 Neuroph 24](#_Toc472842671)

[7.2 Format Sudoku-ID inkl lösung, buchstaben statt ziffern 24](#_Toc472842672)

[7.3 Generator, welcher auf schwierigkeitsstufe zielt 24](#_Toc472842673)

[7.4 In Zelle speichern, zu welchem container sie gehört, -> methoden überarbeiten 24](#_Toc472842674)

[7.5 Zusätzliche Lösungsmethoden, weiteres feature engineering 24](#_Toc472842675)

[7.6 Explizites prüfen der schwierigkeitsstufen 24](#_Toc472842676)

[7.7 Grösseres Trainingsset 24](#_Toc472842677)

[7.8 Generierung von Sudokus visuell 24](#_Toc472842678)

[7.9 Aktuelle Schwierigkeitsstufen mit Rätsel Agentur AG absprechen und evtl. anpassen 24](#_Toc472842679)

[8 Schluss 24](#_Toc472842680)

[9 Literaturverzeichnis 25](#_Toc472842681)

[10 Abbildungsverzeichnis 25](#_Toc472842682)

[11 Anhang 26](#_Toc472842683)

[12 Ehrlichkeitserklärung 26](#_Toc472842684)

# Einleitung

Unser Projekt ist ein Sudokugenerator, welcher neue Sudokus auf der Basis von gegebenen Sudokus mit 17 Ziffern generiert und in Schwierigkeitsstufen kategorisiert. Zur Generierung werden die 17er-Sudokus mehreren verschiedenen Permutationen unterzogen. Danach werden zusätzliche Ziffern aus der Lösung des Sudokus hinzugefügt, was zu einzigartigen Sudokus führt. Die für die Einstufung der Schwierigkeit werden die Sudokus mittels menschlichen Lösungsmethoden gelöst und danach durch ein neurales Netzwerk klassifiziert.

Der Kunde des Projektes ist die Rätsel Agentur AG, welche Rätsel für viele verschiedene Print- und Onlinemedien vertreibt. Unter ihren Kunden befinden sich «20 Minuten» und der «Blick am Abend».

Bisher hat die Rätsel Agentur AG ihre Rätsel von externen Anbietern eingekauft.

Das Ziel unseres Projektes ist ein Sudokugenerator, welcher auf Knopfdruck Sudokus in verschiedenen Schwierigkeitsstufen generiert.

Zuerst werden die Konzepte und Vorgehensweisen bezüglich der einzelnen Komponenten des Projekts erläutert, danach folgt die technische Umsetzung. Zum Schluss werden Resultate und mögliche Erweiterungen des Projektes diskutiert.

# Konzepte

Skalierbarkeit

Idee -> Umsetzung

Erweiterbarkeit/Verbesserungsvorschläge/Änderungsvorschläge-ansätze

## Anwendungsdomäne/

Unser Projekt wird als Informatikprojekt 5 an der Fachhochschule Nordwestschweiz durchgeführt. Das Projekt befindet sich im Kontext eines KTI Projektes des Institutes für 4D Technologien. Das Gesamtprojekt hat ein Webportal zum Ziel, mit welchem sich verschiedene Rätselarten generieren lassen. Ein anderes Studentenprojekt hat sich mit weiteren Logikrätseln befasst.

### Herkunft

<https://www.theguardian.com/media/2005/may/15/pressandpublishing.usnews>

<http://www.sudokudragon.com/sudokuhistory.htm>

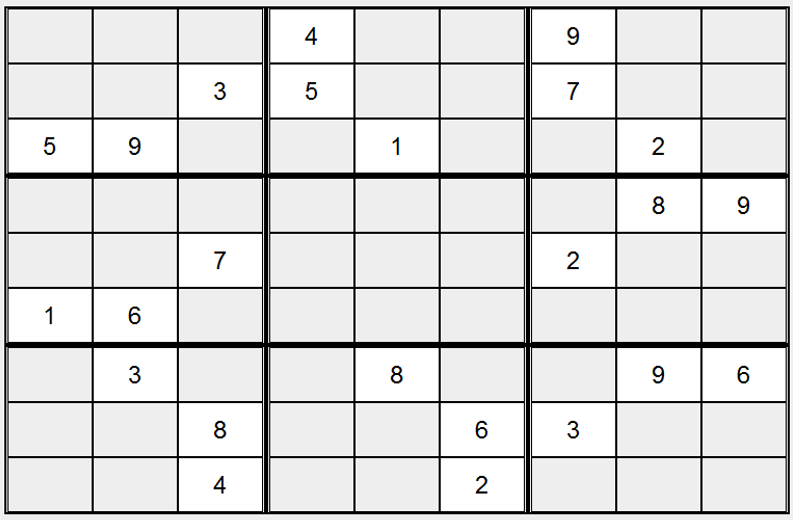
Sudoku ist ein weltweit sehr beliebtes Zahlenrätsel. Da zur Lösung nur Logik und keine komplexen arithmetischen Berechnungen nötig sind, ist es für jeden verständlich.

In seiner Grundform basiert es auf den lateinischen Quadraten des Schweizer Mathematikers Leonhard Euler. Das Heute als Sudoku bekannte Rätsel wurde Ende der 1970er Jahre im New Yorker «Dell Puzzle Magazine» unter dem Namen «Number Place» veröffentlicht. Das Rätsel verbreitete sich ins rätselbegeisterte Japan und wurde dort mit dem Namen «Sudoku» versehen, wobei «Su» Nummer und «Doku» einzeln bedeuten.

Nach mehreren Jahren hoher Popularität im Land der aufgehenden Sonne verbreiteten sich Sudokus mit der Zeit in der ganzen Welt.

Seit 2006 werden jährlich Sudoku-Weltmeisterschaften abgehalten, wobei im Jahr 2016 213 Rätsel-Liebhaber aus 33 verschiedenen Ländern teilgenommen haben.

### Aufbau und Regeln



Spalte

Zeile

Box

Am weitesten verbreitet sind Sudokus, welche aus einem 9x9 Gitter bestehen. Dieses ist zu Beginn mit einigen Ziffern von 1 bis 9 gefüllt.

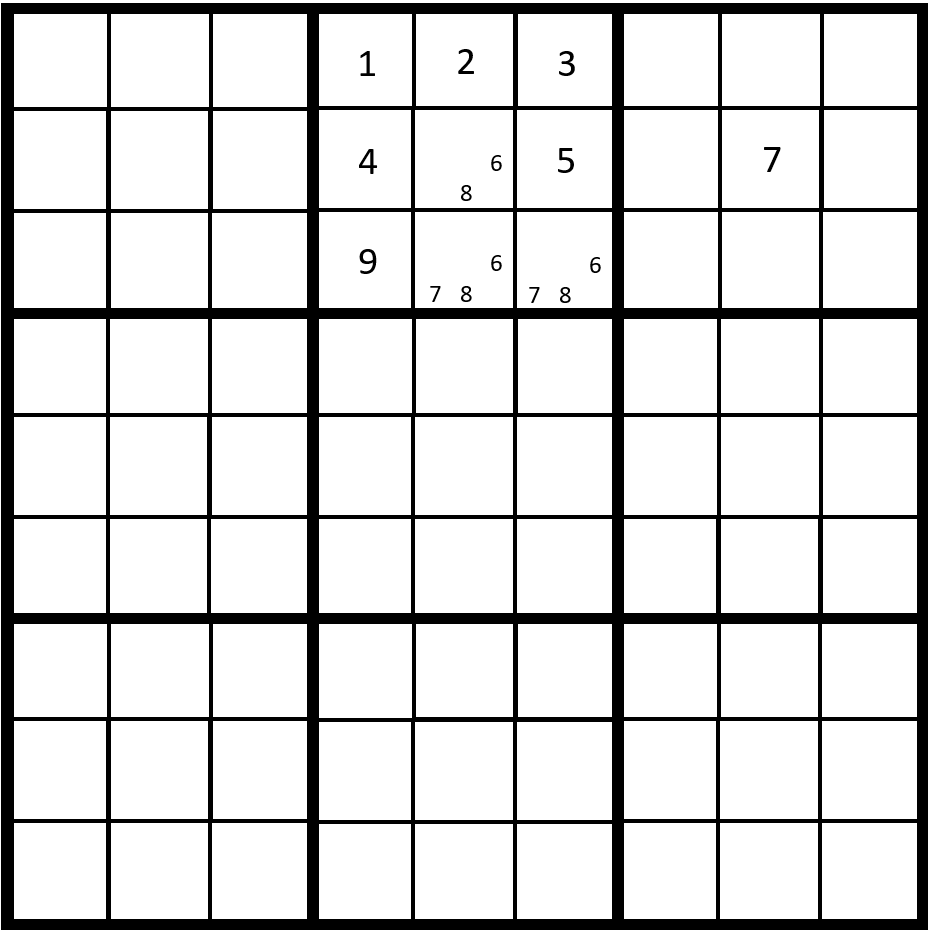
Ziel des Rätsels ist es, das gesamte Gitter mit Ziffern zu füllen, wobei nur eine einzige einfache Regel beachtet werden muss.

In jeder Zeile, Spalte und jeder Box (3x3 Untergitter) muss jede Ziffer von 1 bis 9 genau ein einziges Mal vorkommen.

## Lösen von Sudokus

Zur Lösung von Sudokus haben sich diverse Lösungsmethoden etabliert. Diese Lösungsmethoden sind sogenannte menschliche Lösungsmethoden. Der Begriff menschliche Lösungsmethoden wurde so gewählt, weil diese Techniken von Menschen, welche Sudokus lösen, benutzt werden.

Zum Lösen von Sudokus werden häufig Markierungen (engl. Pencilmarks) verwendet. Pencilmarks sind kleine Notationen pro Zelle, welche die noch möglichen Zahlen für diese Zelle repräsentieren. **Abb.1**



Pencilmarks werden durch das Anwenden diverser Lösungsmethoden immer weiter verringert, bis nur noch eine Markierung in einer Zelle vorhanden ist. Ist dies der Fall, kann der Zelle der Wert der übriggebliebenen Pencilmark zugewiesen werden, da diese Zahl die einzige ist, die in dieser Zelle noch möglich ist.

Lösungsmethoden können in zwei Kategorien unterteilt werden, Techniken die den Zellen einen Wert zuweisen und solche die die Pencilmarks diverser Zellen verringern. Die meisten Menschen wissen nicht, dass ihr Vorgehen beim Lösen eines Sudokus in spezielle Methoden eingeteilt werden kann. Folgend werden jene Lösungsmethoden beschrieben, welche wir in unserer Arbeit verwendet haben.

### Lösungsmethoden welche Werte setzen

#### Naked Single

Ein Naked Single ist eine Zelle die nur noch eine einzige Pencilmark hat und somit der Wert der Zelle auf den der Pencilmark gesetzt werden kann.

Bild

#### Hidden Single

Ein Hidden Single entsteht, falls in einem Container, d.h. in einer Zeile, Spalte oder einer Box ein Wert nur noch in einer einzigen Zelle möglich ist. In diesem Fall kann die Zelle mit jenem Wert ergänzt werden, unabhängig davon wie viele Pencilmarks die Zelle noch hatte.

Bild

### Lösungsmethoden welche Pencilmarks verringern

#### Naked Subset

Ein Naked Subset beschreibt den Fall, wenn innerhalb einer Zeile, Spalte oder Box eine Anzahl von Werten nur in derselben Anzahl von Zellen gesetzt werden können und in diesen Zellen sonst keine Zahlen vorkommen können. Dies hat zur Folge, dass jene Werte in diesen Zellen vorkommen müssen und in allen anderen Zellen der Zeile, Spalte oder Box, jene Zahlen nicht erscheinen dürfen. Folglich können in den restlichen Zellen die Pencilmarks für diese Werte entfernt werden.

Bild

#### Hidden Subset

Ein Hidden Subset liegt vor, falls innerhalb einer Zeile, Spalte oder Box eine Anzahl von Werten nur in genau derselben Anzahl Zellen vorkommen können. Daraus folgt, dass in diesen Zellen zwingend einer Werte gesetzt werden muss, da diese Werte nirgends sonst im jeweiligen Container vorhanden sein können. Somit können für die betroffenen Zellen alle Pencilmarks, die nicht zum gefunden Subset an Werten gehören, entfernt werden.

Bild

#### Block-Line Interactions

Block-Line Interactions, auch «Locked Candidates» genannt, ist eine Technik bei der die Auswirkungen einer Box auf eine Zeile oder Spalte betrachtet werden. Kommen in einer Box alle Pencilmarks für einen Wert auf derselben Zeile oder Spalten vor, so muss dieser Wert in einer diesen Zellen vorkommen und deckt somit den Platz für diesen Wert in der Zeile oder Spalte ab. Deshalb können in allen anderen Zellen der Zeile oder Spalte (die nicht mit der Box überlappen) die Pencilmarks für diesen Wert entfernt werden.

Bild

#### X-Wing

asdf

## Schwierigkeitseinstufung

Die Rätsel Agentur AG hat ihre Sudokus in sieben verschiedene Schwierigkeitsklassen unterteilt. Die Schwierigkeitsstufen lauten folgendermassen:

* Very Easy
* Easy
* Medium
* Hard
* Very Hard
* Very Hard Expert
* Evil/Exotic

Die letzte Schwierigkeitsstufe wird nicht aktiv produziert und muss von uns auch nicht mit dem Generator erreicht werden.

Wir haben diese Schwierigkeitsstufen übernommen und teilen schlussendlich unsere generierten Sudokus in diese Stufen ein.

Um Sudokus in eine Schwierigkeitsklasse einzustufen haben wir Machine Learning verwendet. Mithilfe eines neuralen Netzwerkes beurteilen wir anhand gewissen Eigenschaften, welcher Schwierigkeitsstufe ein gegebenes Sudoku entspricht.

Zu Beginn findet eine Anlernphase statt, in welcher dem neuralen Netzwerk einige Sudokus gezeigt werden, welche bereits in eine Schwierigkeitsklasse eingestuft wurden. Anhand dieser Daten lernt das Netzwerk, was ein Sudoku einer gewissen Schwierigkeit ausmacht.

Als nächstes wird mit weiteren bereits eingestuften Sudokus getestet, wie genau das Netzwerk neue Sudokus in die verschiedenen Schwierigkeitsstufen einteilt.

Sind nach der Testphase die gewünschten Werte erreicht, kann das Netzwerk zur Einstufung neuer Sudokus verwendet werden.

### Modell

Das neurale Netzwerk beurteilt die Sudokus anhand mehrerer Features.

Da nur die Methoden Naked und Hidden Single Zahlen in das Feld einsetzen, werden diese nicht als absolute Werte verwendet. Die Anzahl der jeweils verwendeten Methoden wird mit der gesamten Anzahl der noch fehlenden Zahlen ins Verhältnis gesetzt.

Bei den Naked und Hidden Subsets betrachten wir nur die Subsets, welche eine Grösse von zwei, drei oder vier haben, da ein menschlicher Löser grössere Subsets nur extrem schwer finden kann.

Ebenfalls wird die Anzahl der Verwendungen der anderen beiden Lösungsmethoden Block Line Interactions und X-Wing als Feature im Modell verwendet.

Neben den Lösungsmethoden haben wir weitere Features gewählt, welche auf die Einstufung einen Einfluss haben. Hierzu gehört die Anzahl der vorgegebenen Zahlen, sowie die Anzahl der möglichen Startpositionen, an welcher mittels der Naked oder Hidden Single Methoden als erstes eine Zahl gesetzt werden kann. Für jede der verschiedenen Ziffern wird die Anzahl möglicher Startpunkte als einzelnes Feature verwendet. Als weiteres Feature wird die gesamte Anzahl der Pencilmarks im Sudoku ins Modell hinzugenommen. Die Markierungen zeigen für jede Zelle, welche Ziffern noch nicht ausgeschlossen wurden. Als letztes kommt hinzu, ob das Sudoku mit unseren Lösungsmethoden lösbar war (1) oder ob es mit Backtracking gelöst werden musste (0).

Diese von uns gewählten Features führen zu folgendem Modell:

* Anz. Naked Singles / Anz. fehlende Ziffern
* Anz. Hidden Singles / Anz. fehlende Ziffern
* Anz. Naked Subsets der Grösse zwei bis vier
* Anz. Hidden Subsets der Grösse zwei bis vier
* Anz. Block Line Interactions
* Anz. X-Wing
* Anz. gegebener Ziffern
* Anz. möglicher Startpositionen für jede Ziffer
* Anz. Pencilmarks
* Lösbar mit Lösungsmethoden

## Generierung

Korpus 17er Sudokus

Symmetrien

Lösungen verändern (Spalten/Zeilen/Drehungen)

# Technische Umsetzung

## Technologien

Für die Realisierung unseres Projekts haben wir die Programmiersprache Java gewählt. Das Gesamtprojekt des Institutes für 4D Technologien baut auf der Java Virtual Machine auf und es wurde uns nahegelegt, eine Programmiersprache zu wählen, welche auch auf der JVM zu Hause ist. Wir haben schon viel mit Java gearbeitet und haben so die Einarbeitungszeit in die Grundfeatures der Sprache auslassen können.

Ein weiterer Punkt, welcher für die Wahl von Java spricht sind die vielen Ressourcen, welche im Netz erhältlich sind. Ebenfalls gibt es eine grosse Anzahl an Frameworks, welche Open Source verfügbar sind.

Zur Verwaltung und Versionierung unseres Quellcodes verwenden wir Git.

Das neurale Netzwerk wurde in Matlab traininert und getestet. Für die Umsetzung des Netzwerkes in Java haben wir das Framework Neuroph verwendet.

## Abbildung des Sudoku-Spielfelds

Einfacher String,

Mögliche erweiterung/problematik unseres Formats (Speicherung als einfacher string, keine möglichkeit zu mehr als 10 ziffern, keine unterscheidung zwischen Sudoku und Sudokulösung, kein Index in file usw) Evtl diesen Teil in Resultate auslagern

Board/Row/Column/Box/Cell

Um unser Spielfeld persistent abzulegen, schreiben wir es in ein String, welcher in eine «.sudoku» Datei abgelegt wird.

Der Feldstring ist folgendermassen aufgebaut:

[Feldhöhe] [Feldbreite] [Boxhöhe] [Boxbreite] [Alle Positionen] {Schwierigkeitsstufe}

Die Schwierigkeitsstufe ist nur im String enthalten, wenn diese bereits ermittelt wurde.



Abb. 1: Speicherformat der Sudokus

Updater

## Datensätze

Zum Anlernen des neuralen Netzwerkes haben wir zwei Datenpakete erhalten. Das erste Paket besteht aus Sudokus, welche die Rätsel Agentur AG eingekauft hat, das zweite Paket stammt aus Eigenproduktion. Die erhaltenen Sudokus sind in unterschiedlichen Formaten (siehe Abb. 2 und 3 abgespeichert. Wir haben die Sudokus mittels eines Parsers für das jeweilige Ursprungsformat in unser eigenes «.sudoku» Datenformat (siehe Abb. 1) umgewandelt und so zur weiteren Verwendung abgespeichert.



Abb. 2: Format des ersten Datenpaketes



Abb. 3: Format des zweiten Datenpaketes

### Datenaufbereitung für Statistik

Um Statistiken für die Analyse und Klassifizierung zu erstellen, gehen wir folgendermassen vor. Wir lesen die gewünschten Sudokus in unser Programm ein. Danach wird jedes Sudoku gelöst, wobei die verwendeten Lösungsmethoden für jedes Sudoku gespeichert werden. Ebenfalls werden die anderen benötigten Daten wie die Anzahl der gegebenen Ziffern, die Anzahl der möglichen Startpositionen und die Anzahl der Pencilmarks aus dem Sudoku ausgelesen.

Die Daten aller Sudokus werden danach Komma-getrennt in die gewünschte Datei geschrieben.

## Lösungsmethoden

Erklären mithilfe Board/Row/Column

## Schwierigkeitseinstufung

### Matlab

Die Daten, welche Matlab verwenden soll, müssen aus der generierten Statistik-Datei importiert werden. Weiter werden die importierten Daten zu unseren Features verarbeitet. Einige Features haben eine grosse Varianz, zwischen anderen Features unterscheiden sich die Wertebereiche stark. Um dies abzuschwächen werden die Features mittels Logarithmus in einen ähnlichen Wertebereich gebracht. Wir haben uns gegen eine Normalisierung entschieden, weil diese vom gesamten Datensatz abhängig wäre und wir aber die Möglichkeit bieten wollen, einzelne Sudokus zu klassifizieren.

Die Funktion zur Klassifizierung der Sudokus wurde in einer Anlernphase mit dem ersten Datenpaket mittels der Matlab-App «Neural Net Fitting» erstellt.

Es werden 80% der Sudokus verwendet, um diese Funktion zu erstellen.

In der App wird 5% als Validation und 10% als Test Set verwendet. Wir erstellen ein Netzwerk mit einemeinzigen Hidden Layer, welcher aus 50 Hidden Neurons besteht. Als Trainingsalgorithmus haben wir Levenberg-Marquardt gewählt. Nach der Anlernphase führen wir mit den restlichen 20% der Sudokus einen eigenen manuellen Test durch, um die Genauigkeit zu überprüfen.

Die generierten Sudokus können mithilfe von Matlab klassifiziert werden. Sie müssen wie zu Beginn des Kapitels beschrieben importiert werden.

Die generierten Sudokus werden danach vom trainierten Netzwerk klassifiziert. Die Zuordnung zwischen Sudoku und Schwierigkeitsstufe wird in einer Matrix (results) abgespeichert.

### Neuroph

Es ist nicht benutzerfreundlich, wenn zur Klassifizierung der Sudokus immer eine Statistik-Datei generiert, danach Matlab gestartet und die Daten eingelesen werden müssen.

Aus diesem Grund haben wir unser neurales Netzwerk mithilfe des Frameworks Neuroph in Java abgebildet.

Da wir die genauen angelernten Werte des Netzwerkes nicht direkt aus Matlab übernehmen können, erstellen wir ein neues Netzwerk, welches auf den Erkenntnissen aus Matlab basiert.

Wir verwenden einen MultiLayerPerceptron, welcher auch aus einem Hidden Layer aus 50 Nodes besteht. Zum Trainieren des Netzwerkes werden 80% der Sudokus des ersten Datensatzes verwendet, zum Testen die restlichen 20%. Als Übergangsfunktion haben wir die Sigmoid-Funktion und als Lernalgorithmus MomentumBackpropagation verwendet.

Die Aufbereitung von der Statistik zum endgültigen Feature-Vektor wird auch komplett in Java durchgeführt.

## Generierung

# Resultate

## Lösungsmethoden (gewisse kaum gebraucht)

## Datenanalyse

Vergleicht man die beiden Datenpakete, welche wir erhalten haben, fallen einige gröbere Unterschiede auf.

Der erste Datensatz enthält keine Sudokus der Stufe very hard und der zweite Datensatz keine Sudokus der Stufe evil/exotic. Ebenfalls fällt auf, dass im zweiten Datensatz bereits ab der Schwierigkeitsstufe easy gewisse Sudokus nicht mehr mit unseren Lösungsmethoden gelöst werden können und auf Backtracking zurückgegriffen werden muss. Die Abbildungen 4 und 5 zeigen dies anschaulich auf.



Abb. 4: Datenpaket 1, prozentuale Anzahl gelöster Sudokus



Abb. 5: Datenpaket 2, prozentuale Anzahl gelöster Sudokus

Weiterhin sieht man auf Abbildung 6 gut, dass die durchschnittliche Anzahl der vorgegebenen Ziffern im ersten Datenpaket von very easy bis hard abfällt. Im zweiten Datenpaket (Abb. 7) sieht man jedoch gar keine solche Veränderungen.



Abb. 6: Datenpaket 1, durchschnittliche Anzahl vorgegebener Ziffern



Abb. 7: Datenpaket 2, durchschnittliche Anzahl vorgegebener Ziffern

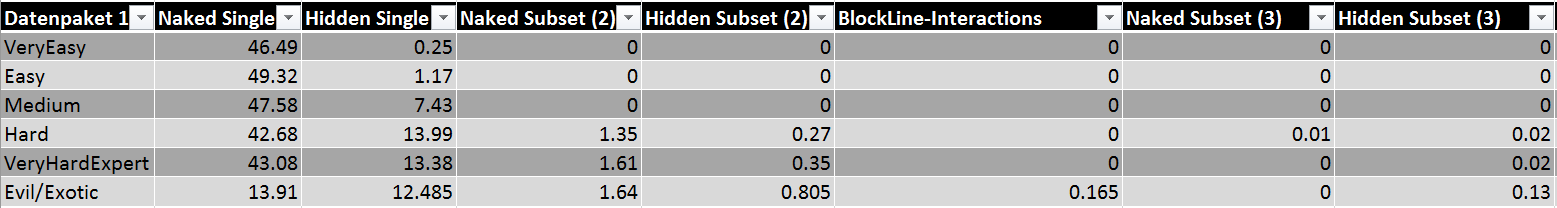
Aufgrund dieser Ungereimtheiten, welche auch beim trainieren des Netzwerkes zu Schwierigkeiten führten, haben wir uns entschieden, nur mit dem ersten Datenpakt zu arbeiten.

Diese Entscheidung hat jedoch zur Folge, dass keine Sudokus der Schwierigkeitsstufe very hard generiert werden, da das erste Datenpaket keine solchen enthält.

## Schwierigkeitseinstufung

Die Rate, mit welcher beim Testen des neutralen Netzwerkes die Sudokus richtig eingeteilt werden zeigt auf, dass die Einstufung mittels menschlicher Lösungsmethoden ein guter Ansatz ist.

Abbildung 8 zeigt, dass die Schwierigkeitsstufen eine starke Auswirkung auf die durchschnittliche Anzahl der verwendeten Lösungsmethoden haben. Daraus folgt, dass die Lösungsmethoden auch für unsere generierten Sudokus ein guter Massstab für die Schwierigkeit sind.



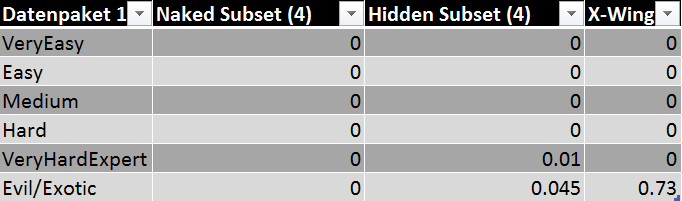


Abb. 8: Durchschnittliche Anzahl Lösungsmethoden pro Schwierigkeitsstufe

Die Genauigkeit Schwierigkeitszuweisung hängt sehr stark mit der Qualität der Trainingsdaten zusammen.

## Matlab/Neuroph

Wir haben uns für die Verwendung der Matlab App «Neural Net Fitting» entschieden, nachdem wir mehrmals viel Zeit damit verbracht hatten, unser Netzwerk an die verschiedenen Feature-Ideen anzupassen.

Diese App erleichtert das Feature Engineering und den daraus folgenden Umbau eines neuralen Netzes stark. Durch diese Applikation konnte bei der Anpassung der Netzwerke einiges an Aufwand eingespart werden.

Mit unserem Netzwerk erreichen wir auf dem ersten Datenpaket eine Genauigkeit von 75%. (Siehe Abb. 8)



Abb. : Konfusionsmatrix basierend auf erstem Datenpaket (Matlab)

Neuroph eignet sich gut, um ein einfaches neurales Netzwerk in Java zu implementieren, da sich schnell ein Netzwerk erstellen lässt. Dabei gilt es aber zu beachten, dass die genauere Konfiguration des Netzwerkes sehr komplex sein kann.

Das Netzwerk, welches wir in Java aufgebaut haben, erreicht aber nicht die Genauigkeit des Matlab-Netzwerkes. (Siehe Abb. 9)

Zusätzlich variieren die Resultate in Neuroph, mit einer Genauigkeit von 61% - 66% und grösseren Abweichungen bei Fehleinschätzungen, stärker als in Matlab.



Abb. : Konfusionsmatrix basierend auf erstem Datenpaket (Neuroph)

## Generator, verteilung der Schwierigkeitsstufen (geschwindigkeit)

## 

Die Einstufung der Schwierigkeit mittels Machine Learning anhand menschlicher Lösungsmethoden hat sich komplexer erwiesen als zuerst angenommen. Die Genauigkeit dieser Einstufung hängt sehr stark mit der Qualität der Trainingsdaten zusammen.

Wir verwenden nur die Daten des ersten Datenpakets als Trainingsgrundlage, da beim zweiten Datenpaket zu viele Ungereimtheiten aufgefallen sind. Da das erste Paket keine Sudokus der Schwierigkeitsstufe 6 enthält, werden Sudokus dieser Stufe, wie auf **Abb. 3** ersichtlich, immer falsch eingestuft. In Abb. 3 werden 20% aller Sudokus aus Datenpaket 1 und 2 durch unser Netzwerk eingestuft.



Abb. 11: Verteilung der Test-Sudokus, welche aus 20% aller Sudokus bestehen

Unsere Lösung versucht momentan, die generierten Sudokus in die Schwierigkeitsstufen des ersten Datenpakets einzuteilen. Dies führt dazu, dass die Einstufung des ersten Datenpakets nachgebildet und nicht eine komplett neue Einteilung vorgenommen wird.

Für eine allgemeinere Einstufung von Sudokus sind mehr Trainingsdaten notwendig, welche auch nachweislich dem Empfinden von Sudokulösern entsprechen.

Ein allgemeiner Test, wie gut die Einstufung von Sudokus dem menschlichen Schwierigkeitsempfinden entspricht ist nicht einfach möglich, da nicht jede Person ein Sudoku gleich beurteilt.

Ein solcher Test kann als weiterführende Massnahme zur Qualität der generierten Sudokus durchgeführt werden. Es muss beachtet werden, dass generierte Sudokus aller verschiedenen Schwierigkeiten mehreren Personen mit möglichst breit gefächertem Sudoku-Vorwissen vorgelegt werden. Es muss beachtet werden, dass auch zwei Personen, welche gleich gut Sudokus lösen, nicht jedes Sudoku gleich beurteilen. Es kann passieren, dass die eine Person eine geeignete Startposition übersieht und danach komplexere Lösungsmethoden anwenden muss, was wiederum dazu führt, dass sie das Sudoku als schwer einstuft.

Um also die Einstufung unserer Lösung zu verifizieren, müssen viele generierte Sudokus gelöst werden. Führt dies zum Resultat, dass unsere Sudokus nicht den Ansichten der Löser entsprechen, so muss mit weiteren Trainingsdaten neu trainiert werden. Die neuen Trainingsdaten müssen von den ersten beiden Datenpaketen unabhängig sein.

Möglichkeit zu unsupervised learning, problem dabei

Library (Neuroph) brauchbar?

## Generierung

17er korpus sinnvoll als basis?

Permutationen hilfreich?

Analyse generierter sudokus (verhältnis generiert : schwierigkeitsstufe, anzahl «nutzloser» sudokus

Evtl. Kommentar zu «unbrauchbaren» sudokus, ansprechen, dass mehr lösungsmethoden auch zu mehr generierten (schwierigen) sudokus führen könnten.

# Erweiterungen

## Verbesserung der Klassifizierung

Um bei der Klassifizierung bessere Resultate zu erzielen können verschiedene Massnahmen ergriffen werden. Die Features, welche unser Modell ausmachen können weiter ausgearbeitet und falls nötig erweitert werden. Dies kann z.B. die Implementierung neuer Lösungsmethoden oder die Einbindung anderer Eigenheiten eines Sudokus sein.

Soll bei der Java Implementation die Genauigkeit erhöht und die Varianz der Resultate verringert werden, kann das Framework anders konfiguriert werden. Dies umfasst beispielsweise die Auswahl eines andern Lernalgorithmus oder einer anderen Übergangsfunktion.

## Format Sudoku-ID inkl lösung, buchstaben statt ziffern

Um Sudokus einfacher zu unterscheiden und zu identifizieren kann aus den Zellen der Lösung des Sudokus ein Hashwert generiert werden, welcher dann zur Identifizierung der Sudokus verwendet wird.

## Generator, welcher auf schwierigkeitsstufe zielt

Sollen nur Sudokus einer bestimmten Schwierigkeit generiert werden, kann das Programm so erweitert werden, dass alle Sudokus, welche nicht der Schwierigkeit entsprechen wieder verworfen werden. Dadurch wird jedoch die Zeit, welche für das Generieren der Sudokus benötigt wird steigen, da insgesamt mehr Sudokus generiert werden.

## In Zelle speichern, zu welchem container sie gehört, -> methoden überarbeiten

Soll die Performance der Updater Klasse erhöht werden, kann das Programm so erweitert werden, dass in jeder Zelle gespeichert wird, zu welcher Box, Spalte und Zeile sie gehört. Dies führt auch zu einer Vereinfachung der Implementation gewisser Lösungsmethoden.

## Validierung der generierten Schwierigkeitsstufen

Um sicher zu gehen, dass die Einstufung der generierten Sudokus korrekt ist, gibt es mehrere mögliche Vorgehensweisen. Eine Idee ist eine zusätzliche Einordnung mit einem externen Klassifizierer, welcher bereits für die Einstufung von Sudokus verwendet wird.

Eine weitere Möglichkeit ist eine Feldstudie mit Personen, welche über möglichst unterschiedliche Sudoku-Kenntnisse verfügen. Diese Personen müssen einige Sudokus lösen und festhalten, wie schwer sie die gelösten Sudokus empfunden haben.

Mithilfe der auf diese Art klassifizierten Sudokus kann das neurale Netzwerk neu trainiert werden um die Genauigkeit zu erhöhen.

## Generierung von Sudokus visuell

Um die Sudokus für Print- und Onlinemedien darzustellen muss eine Erweiterung für das Programm implementiert werden, welche aus dem Feldstring oder sogar direkt aus dem Board Objekt eine visuelle Darstellung des Sudokus generiert.

## Speicherung der verwendeten Lösungsmethoden pro Zelle

Besteht der Wunsch, für jede Zelle zu visualisieren oder auszuwerten, durch welche Lösungsmethode die Pencilmarks oder der Wert selbst verändert wurde, muss man die verwendeten Lösungsmethoden für jede Zelle separat abspeichern.

# Schluss

Die Einteilung von Sudokus in bestimmte Schwierigkeitsstufen aufgrund menschlicher Lösungsmethoden ist ein guter Ansatz. Dabei muss jedoch notiert werden, dass die komplexeren Lösungsmethoden für unsere generierten Sudokus nur sehr selten verwendet werden. Dazu kommt, dass nicht nur die verwendeten Lösungsmethoden Einfluss auf die Klassifizierung von Sudokus haben.

Es ist sehr wichtig, dass die Klassifizierung der Testdaten von hoher Qualität ist, ansonsten folgt daraus eine schlechte Einteilung der neu generierten Sudokus. Ein grosses Trainingsset ist ebenfalls wichtig, wobei darauf geachtet werde muss, dass alle Sudokus vom selben Klassifizierer eingestuft wurden.

Die drei implementierten Generatormethoden weisen eine unterschiedliche Verteilung der Schwierigkeitsstufen der erstellten Sudokus auf. Weiterhin unterscheiden sich die generierten Sudokus im Symmetriegrad. So kann weiterführend die gewünschte Schwierigkeit und Symmetrie die Wahl des Generators beeinflussen.

Unsere Rolle in diesem Projekt ist nun abgeschlossen und das Projektziel der Implementation eines Sudokugenerators, welcher auf Basis von 17er Sudokus neue Rätsel verschiedener Schwierigkeiten erstellt wurde erreicht.

Unser Projekt wird nun in das KTI-Projekt eingebunden und dort als Teil der gesamten Software weiterentwickelt.

Notwendig.

Analog zur Einleitung soll eine wissenschaftliche Arbeit, ein technischer Bericht einen abrundenden Abschluss

aufweisen. Der Schluss ist nicht nur mit den Schlussfolgerungen gleichzusetzen; er kann das Thema zusätzlich

auch in einen grösseren Zusammenhang stellen, weiterführende Perspektiven aufzeigen, das weitere Vorgehen

skizzieren oder in Form einer Diskussion oder einer Reflexion ein Fazit vorlegen. Seine Hauptaufgabe besteht

indessen zentral darin, die Ergebnisse klar und prägnant zusammenzufassen. Dabei sollen Einleitung und

Schluss zusammengelesen werden können. Sie bilden eine Klammer.

Zusammenfassung **was** Sie **wie** erreicht haben.

Was sind Ihre Erkenntnisse aus der Arbeit?

Vergleich mit der Aufgabenstellung, Projektklärung.

Potentielle Erweiterungen, Vorschläge für nächste Schritte.

Was würden Sie in der nächsten Version anders machen?

Erklärungen, warum zum Beispiel Teilziele nicht erreicht wurden (z.B.

falsche Zeiteinschätzung).

**Achtung**: Persönliche Erfahrungen und Danksagungen gehören nicht

hierhin, sondern, wenn überhaupt, in ein Vorwort.

Viele einflüsse auf einstufung

Höhere Lösungsmethoden für «normale» sudokus unwichtig

Evtl. Zf von erweiterungen

Probleme Generierung

Machine Learning bastelei

# Literaturverzeichnis

Notwendig.

Das Literaturverzeichnis ist Teil des Hauptteils, folgt nach dem Schluss (siehe hier 3.3) und enthält in

alphabetischer Reihenfolge die vollständigen bibliographischen Angaben zur gesamten benutzten Literatur.

Nach der IEEE-Methode fällt das alphabetisch aufgeführte Literaturverzeichnis zugunsten der am Schluss in

eckigen Klammern chronologisch referenzierten Quellen weg. Mehr zu Inhalt und Gestaltung von einzelnen

Literaturhinweisen im Kapitel 6.

# Abbildungsverzeichnis

[Abb. 1: Speicherformat der Sudokus 16](#_Toc472699998)

[Abb. 2: Format des ersten Datenpaketes 16](#_Toc472699999)

[Abb. 3: Format des zweiten Datenpaketes 17](#_Toc472700000)

[Abb. 4: Datenpaket 1, prozentuale Anzahl gelöster Sudokus 17](#_Toc472700001)

[Abb. 5: Datenpaket 2, prozentuale Anzahl gelöster Sudokus 18](#_Toc472700002)

[Abb. 6: Datenpaket 1, durchschnittliche Anzahl vorgegebener Ziffern 18](#_Toc472700003)

[Abb. 7: Datenpaket 2, durchschnittliche Anzahl vorgegebener Ziffern 18](#_Toc472700004)

[Abb. 8: Verteilung der Test-Sudokus, welche aus 20% aller Sudokus bestehen 20](#_Toc472700005)

# Anhang

Fakultativ, in naturwissenschaftlichen oder technischen Arbeiten jedoch oft zweckdienlich und erforderlich, in

umfangreichen Projektarbeiten oft unumgänglich.

Der Anhang enthält Zusatzmaterialien, die für bestimmte Zielgruppen relevant sind. Alles, was für das

Verständnis des Textes unmittelbar notwendig ist, gehört in den Text selbst. Ein Anhang kann Skizzen, Pläne,

detaillierte Berechnungen, Laborauswertungen, Interviewprotokolle, Fragebogen, Übersichtstafeln, Glossare,

Programmcode, Statistiken etc. enthalten.

Das Projektmanagement kann ebenfalls im Anhang Platz finden. Der Anhang kann also Verträge (Lasten- und

Pflichtenheft), Projektpläne mit Meilensteinen, sowie Zeiterfassungen und dergleichen enthalten.

Teil des Anhangs können auch Anleitungen für Personen sein, die ein Gerät benützen, montieren, warten. Also:

Gebrauchsanleitung, Montageanleitung, Wartungsanleitung.

Alle wichtigen Informationen gehören in den ausgedruckten Anhang. Die unwichtigeren oder zu umfangreiche

Informationen, beispielsweise Software-Code, sollen nicht ausgedruckt werden; sie gehören auf ein Speichermedium,

beispielweise auf eine DVD, oder können auf einem Laufwerk der Schule oder Server einer Firma

abgelegt werden – das ist unumgänglich bei grossen Datenmengen.

# Ehrlichkeitserklärung

Notwendig.

Schreibende bestätigen mit ihrer Unterschrift, dass sie die Arbeit ohne fremde Hilfe und unter Einhaltung der

gebotenen Regeln erstellt haben. Meist wird dazu folgender Standardsatz verwendet:

«Hiermit erkläre ich, die vorliegende x2 selbständig, ohne Hilfe Dritter und nur unter Benutzung der

angegebenen Quellen verfasst zu haben.»

Dieser Satz ist mit der handschriftlichen Unterschrift sowie Angaben zu Ort und Datum zu begleiten.