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](#_Toc472776364)

[2 Inhaltsverzeichnis 4](#_Toc472776365)

[3 Einleitung 6](#_Toc472776366)

[4 Konzepte 7](#_Toc472776367)

[4.1 Anwendungsdomäne/Umfeld 9](#_Toc472776368)

[4.1.1 Herkunft 9](#_Toc472776369)

[4.1.2 Aufbau und Regeln 11](#_Toc472776370)

[4.2 Lösen 12](#_Toc472776371)

[4.2.1 Naked Single 13](#_Toc472776372)

[4.2.2 Hidden Single 14](#_Toc472776373)

[4.2.3 Naked Subset 15](#_Toc472776374)

[4.2.4 Hidden Subset 16](#_Toc472776375)

[4.2.5 Block-Line Interactions 17](#_Toc472776376)

[4.2.6 X-Wing 18](#_Toc472776377)

[4.3 Schwierigkeitseinstufung 19](#_Toc472776378)

[4.3.1 Modell 19](#_Toc472776379)

[4.4 Generierung 21](#_Toc472776380)

[5 Technische Umsetzung 21](#_Toc472776381)

[5.1 Technologien 21](#_Toc472776382)

[5.2 Abbildung des Sudoku-Spielfelds 21](#_Toc472776383)

[5.3 Datensätze 22](#_Toc472776384)

[5.4 Lösungsmethoden 23](#_Toc472776385)

[5.5 Schwierigkeitseinstufung 24](#_Toc472776386)

[5.5.1 Ausarbeiten der Features 24](#_Toc472776387)

[5.5.2 Verwendetes Netzwerk 24](#_Toc472776388)

[5.6 Generierung 24](#_Toc472776389)

[6 Resultate 24](#_Toc472776390)

[6.1 Lösungsmethoden 24](#_Toc472776391)

[6.2 Schwierigkeitseinstufung 25](#_Toc472776392)

[6.3 Generierung 26](#_Toc472776393)

[7 Schluss 26](#_Toc472776394)

[8 Literaturverzeichnis 27](#_Toc472776395)

[9 Anhang 28](#_Toc472776396)

[10 Ehrlichkeitserklärung 29](#_Toc472776397)

# 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

Notwendig.

Jeder wissenschaftliche Text oder technische Bericht muss eine sinnvolle innere Gliederung aufweisen. Eine

entsprechende Ordnung in einen Stoff zu bringen gehört zu den Herausforderungen, vor denen Schreibende

und Dokumentierende immer wieder von neuem stehen. Jeder Stoff erfordert eine spezifische Ordnung oder

Struktur. Auf keinen Fall soll der Hauptteil erzählende Teile enthalten, also keine Vorgänge schildern, sondern

Ergebnisse beschreibend und objektiv festhalten. Das können zum Beispiel Rechercheergebnisse, Analyseergebnisse,

Konzeptergebnisse, Umsetzungsergebnisse, Test- oder Messergebnisse sein. Der Hauptteil soll ausserdem

leicht nachvollziehbar, transparent und ausgewogen gegliedert sein, das gilt insbesondere für den Aufbau von

Experimenten, für die gewählten Konzepte und methodischen Lösungswege. Die gewonnenen Resultate

müssen diskutiert, d. h. kommentiert und interpretiert werden.

Auf zu viele Gliederungsebenen ist zu verzichten. Ein Unterkapitel soll mindestens einen Abschnitt enthalten.

Hauptkapitel beginnen auf einer neuen Seite.

**Theoretischer Teil**

Mögliche Kapitel (nicht vollständig). Dies sieht je nach Text anders aus:

Beschreibung des Umfelds, bzw. der Anwendungsdomäne

Stand der Forschung/“state of the art“ (Ansätze, Theorien, Modelle)

Literaturstudien

Beschreibung der verwendeten Methoden

Vergleichbare Arbeiten

Feldstudien, Umfragen

Folgende Fragen können hilfreich sein:

Welche ähnlichen Probleme gibt es?

Wie lösen andere das Problem?  
Was ist die Anwendungsdomäne? Beschreiben Sie diese so, dass

Aussenstehende sie auch verstehen.

Nach welchen Methoden sind sie vorgegangen beim Lösen des Problems?

**Praktischer Teil**

Je nach Aufgabenstellung sieht dieser Teil sehr verschieden aus. Hier einige

Fragen:

Wie ist Ihre Lösung aufgebaut (Architektur)?

Gibt es Teile die speziell sind und genau dokumentiert werden müssen

(z.B. UML-Diagramme)?

Performancemessungen, falls vorhanden.

Wie ist die Skalierbarkeit der Lösung?

Haben Sie Daten erfasst? Wie sehen diese im Detail aus?

Welche Punkte müssen bei einer allfälligen Erweiterung berücksichtigt

werden?

Wie sind Sie bei der Lösung vorgegangen (falls das überhaupt von

Interesse ist, z.B. im Interface-Design)?

Welche Alternativen haben Sie geprüft? Weshalb haben Sie sich für die

aktuelle Lösung entschieden?

Benutzerhandbuch, falls notwendig (aber besser im Anhang).

Skalierbarkeit

Idee -> Umsetzung

Erweiterbarkeit/Verbesserungsvorschläge/Änderungsvorschläge-ansätze

## Anwendungsdomäne/Umfeld

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. Zurzeit ist ein weiteres Studentenprojekt im Gange, welches sich mit den Logikrätseln Nonogramm und Hashi auseinandersetzt.

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

Bisher hat die Rätsel Agentur AG ihre Rätsel von externen Anbietern eingekauft. Das Ziel unseres Projektes ist es, ein Programm zu erstellen mit welchem die Rätsel Agentur AG zukünftig Sudokus selber generieren kann und die Sudokus nicht mehr einkaufen muss. Um Sudokus für ein Rätselheft generieren zu können müssen zwei Aufgaben erfüllt werden, erstens müssen neue Sudokus erstellt werden und zweitens müssen die Sudokus nach ihrer Schwierigkeit eingestuft werden können.

### 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.

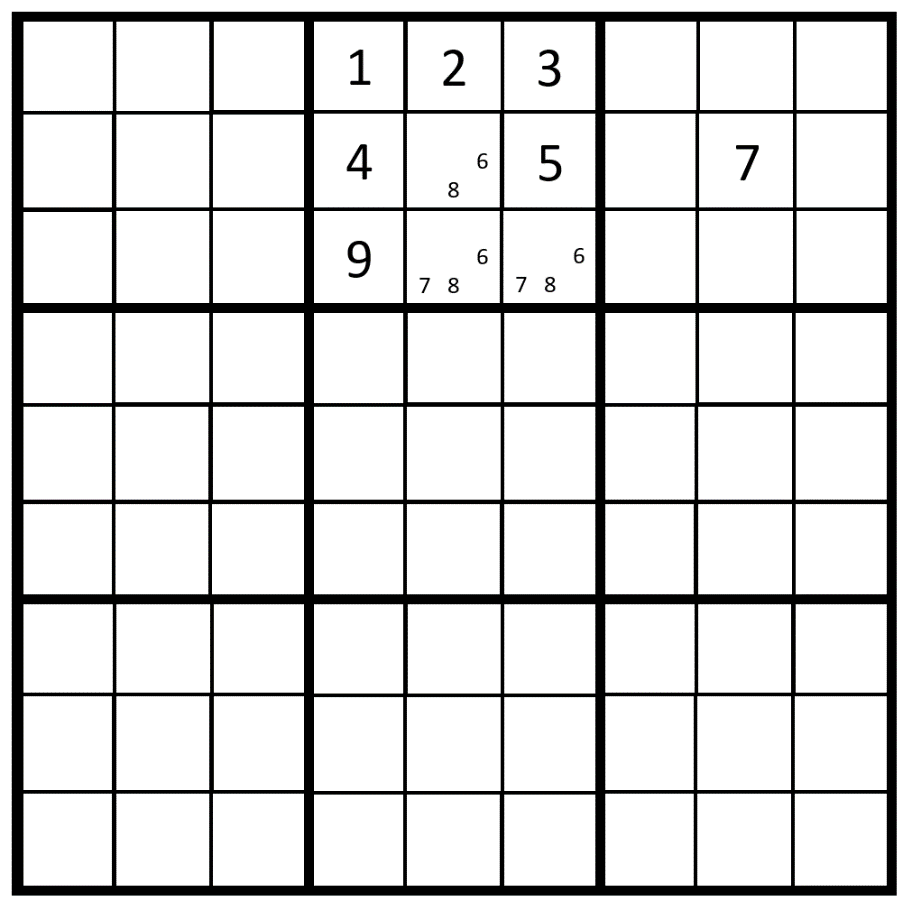
Seit 2006 werden jährlich Sudoku-Weltmeisterschaften abgehalten, wobei im Jahr 2016 213 Rätsel-Liebhaber aus 33 verschiedenen Ländern teilgenommen haben.

### Aufbau und Regeln

Sudokus bestehen meistens aus einem 9x9 Gitter, welches zum Teil mit Ziffern von 1 bis 9 gefüllt ist. Es gibt auch Sudokus in anderen Grössen. Für alle Grössen gelten jedoch dieselben Regeln. 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) darf jede Ziffer von 1 bis 9 nur genau ein einziges Mal vorkommen. 3 Boxen die horizontal oder vertikal nebeneinander stehen bezeichnen wir als horizontalen respektive vertikalen Block.

## 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.

### 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.



Im Beispiel hat die Markierte Zelle nur noch eine einzelne Pencilmark (die Vier) übrig. Somit kann in diesem Feld die Vier gesetzt werden.

### 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 Zelle möglich ist. In diesem Fall kann die Zelle auf diesen Wert gesetzt werden, unabhängig davon wie viele Pencilmarks die Zelle noch hatte.

In diesem Beispiel kann innerhalb der mittleren Box die Sechs nur in der markierten Zelle gesetzt werden. Folglich kann die Zelle mit der Sechs gefüllt werden.

### 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.****

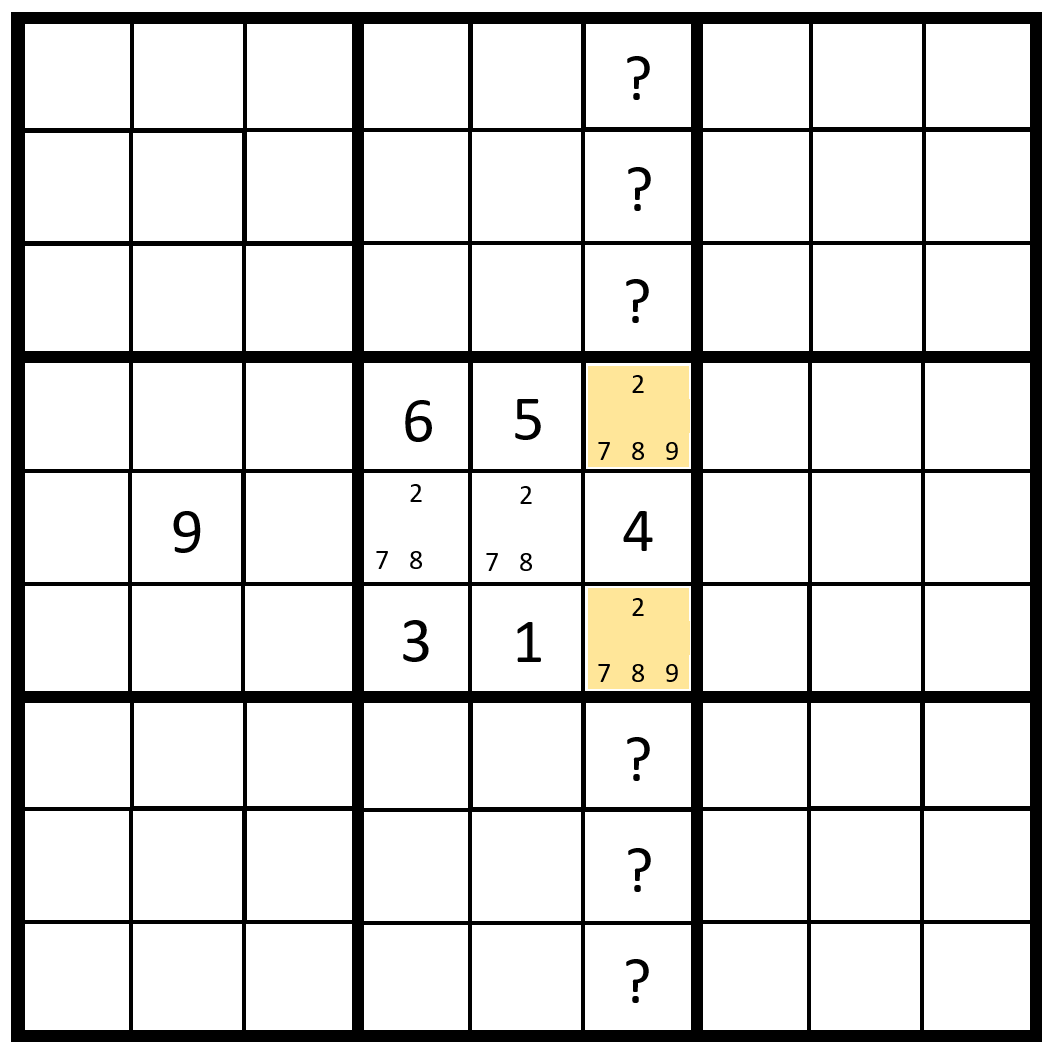
Im Bild ist ersichtlich, dass die Ziffern Zwei und Vier in den markierten Zellen vorkommen müssen. Aus diesem Grund können die beiden Ziffern nirgends in der mittleren Box vorkommen, das heisst, die Pencilmarks für Zwei und Vier können bei den restlichen Zellen der Box entfernt werden.

### 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 der 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.

Die Werte Acht und Neun können in der mittleren Box nur in den markierten Zellen vorkommen. Dies impliziert, dass falls eine der beiden Zahlen in einer Zelle gesetzt wird, die andere Zahl in der übriggebliebenen Zelle vorkommen muss. Somit können die Pencilmarks für die anderen Zahlen in diesen beiden Zellen entfernt werden.

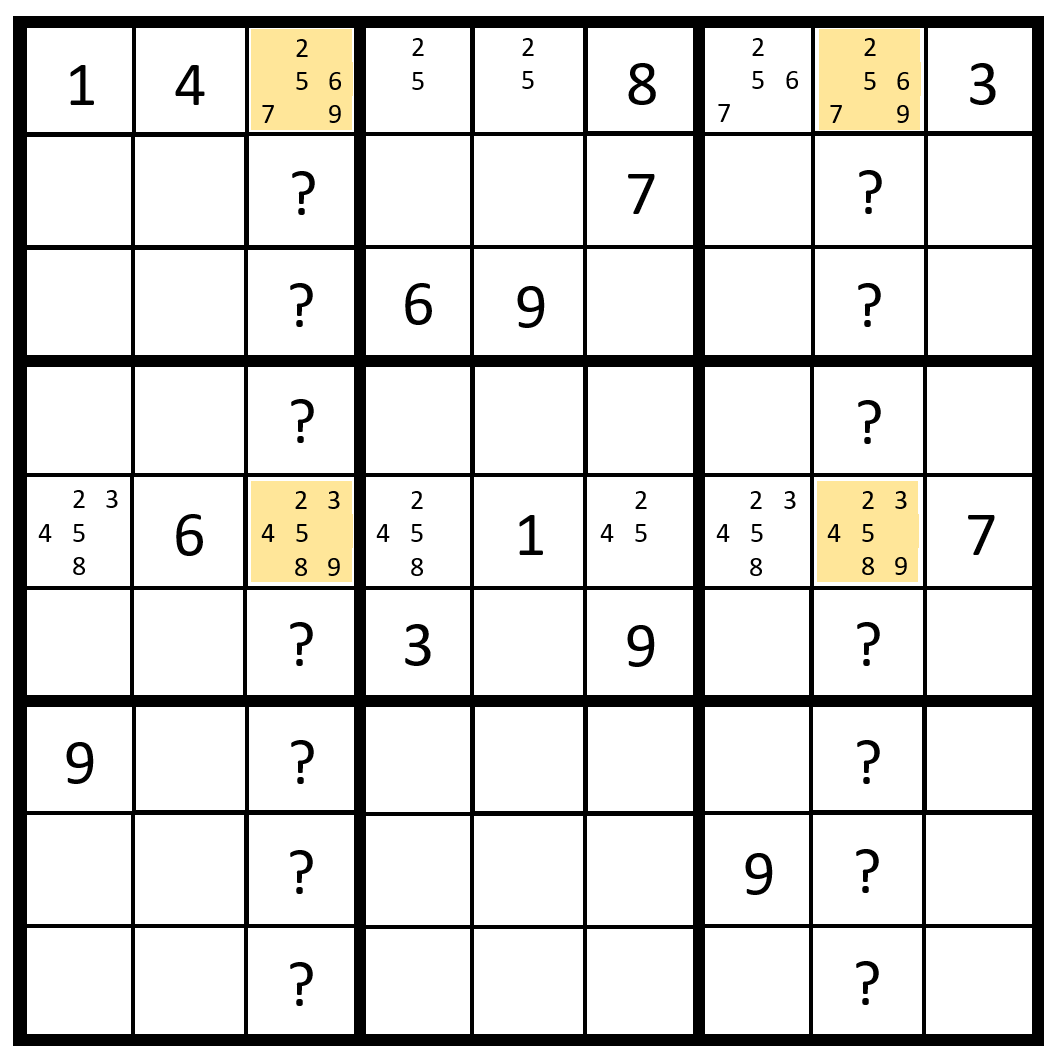
### Block-Line Interactions

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.

In der mittleren Box kann die Zahl Neun nur in den beiden markierten Zellen vorkommen. Da die beiden Zellen beide Teil derselben Spalte sind, kann in den restlichen Feldern der Spalte (mit ‘?’ markiert) die Neun nicht mehr vorkommen und kann deshalb aus den Pencilmarks entfernt werden.

### X-Wing

Ein X-Wing entsteht falls ein Wert in zwei unterschiedlichen Zeilen jeweils genau zweimal vorkommen kann und diese vier Zellen zusammen die Eckpunkte eines hypothetischen Rechteckes bilden. Diese Ausgangslage führt dazu, dass der Wert in diesen zwei Zeilen immer diagonal vorkommt und somit beide Spalten mit dem Wert gefüllt werden. Somit können für alle anderen Zellen, in den zwei Spalten, die nicht einer der vier Zellen entsprechen die Pencilmark für den gewählten Wert entfernt werden.



Im Beispiel kann die Zahl Neun in den Zeilen Eins und Fünf jeweils genau zweimal vorkommen. Da die möglichen Vorkommnisse spaltenweise genau untereinander sind, bilden sie die Eckpunkte eines Rechteckes (markierte Zellen). Das heisst, dass die Neun entweder oben links und unten rechts oder oben rechts und unten links vorkommen kann. In beiden Fällen sind alle betroffenen Spalten mit einer Neun ergänzt worden. Deshalb können in den restlichen Zellen der beiden Spalten (mit ‘?’ markiert) alle Pencilmarks für die Zahl Neun entfernt werden.

## 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 direkt 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 den noch fehlenden Zahlen ins Verhältnis gesetzt. Dies führt dazu, dass die Anzahl der zu setzenden Zahlen keinen Einfluss auf dieses Feature hat.

Wir beachten nur die Naked und Hidden 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.

Wir haben uns für folgendes Modell entschieden.:

* Verhältnis von mittels der Naked Single Methode gesetzten Zahlen zu allen im Sudoku fehlenden Zahlen.
* Verhältnis von mittels der Hidden Single Methode gesetzten Zahlen zu allen im Sudoku fehlenden Zahlen.
* Anzahl benutzter Naked Subset Methoden für zweier bis vierer Subsets.
* Anzahl benutzter Hidden Subset Methoden für zweier bis vierer Subsets.
* Anzahl benutzter Block Line Interaction Methode.
* Anzahl benutzter X-Wing Methode.
* Anzahl gesetzter Zahlen im Sudoku.
* Anzahl möglicher Startpositionen für die Naked und Hidden Single Methoden, wobei für jede einzelne Ziffer ein eigenes Feature erstellt wurde.
* Anzahl möglicher Orte, an welchen eine Ziffer gesetzt werden kann. (Pencilmarks)
* Ob das Sudoku mit den implementierten menschlichen Lösungsmethoden vollständig gelöst werden konnte.

Um neue Features auszuarbeiten benutzen wir die «Neural Network Fitting» App von Matlab.

Von der Rätsel Agentur AG haben wir zwei Blöcke von bereits klassifizierten Sudokus als Testdaten erhalten.

Die Testdaten werden von einem Java-Programm aufbereitet und als Statistics.csv exportiert.

## Generierung

Ein Sudoku muss eindeutig lösbar sein. Das heisst, es darf für ein Sudoku nur eine mögliche Lösung geben. Die Anzahl Lösungswege sind dabei jedoch nicht beschränkt. In einem Sudoku äussert sich diese Eigenschaft dadurch, dass die Reihenfolge von Lösungsmethoden die nicht voneinander abhängig sind keinen Einfluss auf die Lösung des Sudokus haben. Auch die Wahl der Lösungsmethode, sofern anwendbar, kann das Ergebnis nicht verändern.

Die minimale Anzahl Startpositionen, die ein Sudoku haben muss, damit es eindeutig lösbar ist, ist nicht offiziell bekannt. Es gibt jedoch eine Arbeit von Gary McGuire in welcher er durch eine erschöpfende Suche zeigt, dass es keine Sudokus mit 16 oder weniger Startpositionen gibt, die eindeutig lösbar sind. [https://arxiv.org/abs/1201.0749] Diese Arbeit wurde jedoch bis dato nicht publiziert oder von anderen Forschern geprüft.

Damit wir bei der Generierung von neuen Sudokus nicht jeweils noch überprüfen müssen, ob ein Sudoku eine eindeutige Lösung hat, generieren wir unsere Sudokus nicht von Grund auf neu. Wir verwenden stattdessen bereits bekannte Sudokus mit 17 Startpositionen und eindeutiger Lösung (17er Sudoku) als Basis für die Generierung.

### Transformationen

Aus einem 17er Sudoku können mithilfe von Transformationen eine grosse Menge weiterer 17er Sudokus generiert werden. Diese Sudokus sehen unter Umständen sehr verschieden aus, sind jedoch lösungstechnisch genau gleich lösbar wie das originale 17er Sudoku. Denn es werden keine Pencilmarks verändert beim transformieren des Sudokus. Die folgenden Transformationen veränderen nur das Aussehen, jedoch nicht das Lösungsvorgehen des Sudokus.

* Transponieren

Alle Zeilen mit allen Spalten des Sudokus tauschen, dies entspricht einer Spiegelung des Sudokus an einer Diagonalen.

(2 Möglichkeiten)

* Permutationen der 9 verschiedenen Zahlen (9! Möglichkeiten)

Die einzelnen Zahlengruppen untereinander vertauschen, z. B. werden alle Drei zu Vier, alle Vier zu Eins und alle Eins zu Drei.

(9! Möglichkeiten)

* Zeilen innerhalb eines horizontalen Blocks vertauschen

Die 3 Zeilen innerhalb eines horizontalen Blocks beliebig miteinander vertauschen.

(3 x 6 Möglichkeiten)

* Spalten innerhalb eines vertikalen Blocks vertauschen

Die 3 Spalten innerhalb eines vertikalen Blocks beliebig miteinander vertauschen.

(3 x 6 Möglichkeiten)

* Horizontale Blocks vertauschen

Die 3 Horizontalen Blocks beliebig miteinander vertauschen.

(6 Möglichkeiten)

* Vertikale Blocks vertauschen

Die 3 vertikalen Blocks beliebig miteinander vertauschen.

(6 Möglichkeiten)

Kombiniert man diese Transformationen miteinander kann man aus einem Sudoku bis zu 1.2 Billionen verschiedene Sudokus erzeugen. Rotieren und Spiegeln des Sudokus können als Kombination der obenstehenden Transformationen realisiert werden und müssen deshalb nicht speziell aufgelistet werden.

### Ergänzen

Da wir nicht nur 17er Sudokus generieren sollen, ergänzen wir die Sudokus in einem letzten Schritt mit weiteren Zahlen. Da das Sudoku eine eindeutige Lösung hat, ist es nicht möglich zufälligerweise Zahlen dem Feld hinzuzufügen, ohne in die Gefahr zu kommen das Sudoku unlösbar zu machen. Stattdessen wird das Sudoku gelöst und anschliessend aus der Lösung Zahlen ausgewählt und dem Feld hinzugefügt. Mit dieser Methode wird sichergestellt, dass keine falschen Zahlen dem Sudoku hinzugefügt werden.

Um für den Leser das Sudoku visuell ansprechend zu gestalten, achten wir darauf, dass das Sudoku eine möglichst symmetrische Anordnung der Startzahlen aufweist. Im Idealfall ist ein Sudoku horizontal, vertikal sowie über beide Diagonalen symmetrisch und zusätzlich noch Punktsymmetrisch.

Welche Schwierigkeit das Sudoku hat wird bei der Generierung nicht beachtet.

# 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.

Für das Wählen der Features für die Modelle verwenden wir Matlab und die Neural Network Toolbox. Die Umsetzung des Netzwerkes in Java ist mit dem Neuroph Framework implementiert.

## Abbildung des Sudoku-Spielfelds

Für die Repräsentation des Sudoku-Spielfelds haben wir eine Board Klasse erstellt. Diese Klasse beinhaltet ein zweidimensionales Array mit den einzelnen Zellen sowie zusätzlich 3 Arrays für die Zeilen Spalten und Boxen. Jede dieser Container-Instanzen besitzt jeweils ein Array in dem die einzelnen Zellen referenziert werden. Es kann somit über die Container als auch direkt über die Position auf eine Zelle zugegriffen werden.

Um Zellen mit einem Wert zu versehen haben wir eine Klasse Updater definiert, die mithilfe einer statischen Methode den Wert in die Zelle einsetzt und zusätzlich alle betroffenen Pencilmarks aktualisiert. Dabei wird sogleich in der Zelle gespeichert mit welcher Lösungsmethode der Wert gesetzt wurde.

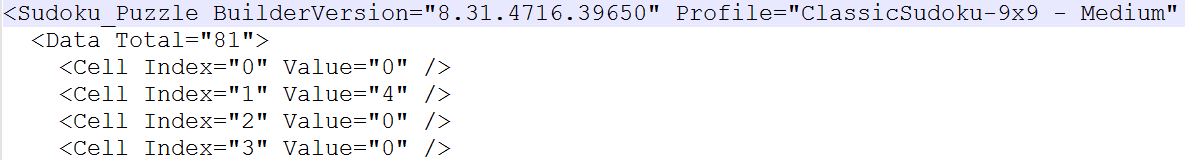
Um unser Spielfeld persistent abzulegen, schreiben wir es in ein String, welcher in eine «.sudoku» Datei abgelegt wird. Der Feldstring ist folgendermassen aufgebaut:

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

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

## Datensätze

Zum Anlernen des neuralen Netzes haben wir zwei Datensatz-Pakete erhalten. Das erste Paket besteht aus Sudokus, welche die Rätsel Agentur AG eingekauft hat, das zweite Paket stammt aus Eigenproduktion.







Vergleicht man die beiden Pakete, 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 mittels Backtracking gelöst werden müssen.



Abb. : Datensatz 1, prozentuale Anzahl gelöster Sudokus



Abb. : Datensatz 2, prozentuale Anzahl gelöster Sudokus

## Schwierigkeitseinstufung

### Ausarbeiten der Features

Um dem neuralen Netzwerk das Einstufen der Sudokus in die verschiedenen Schwierigkeitsstufen beizubringen werden zuerst alle Sudokus gelöst. Dabei wird für jedes Sudoku gespeichert, wie oft welche Lösungsmethode zum Einsatz kam und ob das Sudoku zum Schluss mittels Backtracking gelöst werden musste.

Diese Daten werden zusammen mit allen anderen Features in ein .csv-File exportiert, welches im Matlab wieder in eine Matrix importiert wir, welche zum Anlernen und Testen des Netzwerkes verwendet wird.

Sobald das Modell steht, wird das Netzwerk mit dem ausgewählten Modell in Java mit dem Neuroph-Framework aufgebaut.

### Verwendetes Netzwerk

Unser neurales Netzwerk bildet die 22 Features eines Sudokus auf die sieben Schwierigkeitsstufen ab. Das Netzwerk besteht aus einem Hidden Layer, welcher 10 Nodes enthält. Die gesamten Daten werden normalisiert, bevor das Netzwerk damit arbeitet, dazu wird in Neuroph ein MaxNormalizer verwendet. Als Activation Function verwendet das Netzwerk die logistische Sigmoid-Funktion. Die Kosten werden per Backpropagation optimiert.

## Generierung

Für die Generierung eines Sudokus haben wir einen linearen Ablauf implementiert. Dabei wird ein Sudoku nach dem anderen generiert. Sudokus welche unser Programm nicht ohne backtracking lösen kann werden verworfen, da sie der höchsten Schwierigkeit (evil / exotic) entsprechen würden, die von der Rätsel Agentur AG nicht verwendet werden. Bei der Generierung kann eingestellt werden in welchem Bereich die Anzahl Startzahlen sich befinden soll und wie viele Sudokus generiert werden sollen.

### Basis

Um möglichst verschiedene Sudokus generieren zu können haben wir den Sudoku-Korpus von Gordon Royle als Basis verwendet. [<http://staffhome.ecm.uwa.edu.au/~00013890/sudokumin.php>] Dieser Korpus besteht aus einer Sammlung von Sudokus mit 17 Startpositionen und eindeutiger Lösung, die zudem unter den möglichen Transformationen verschieden sind. Das heisst, dass auch mithilfe der Transformationen kein 17er Sudoku aus dem Korpus auf ein anderes abgebildet werden kann. Der Korpus umfasst fast 50'000 solcher Sudokus.

Für die Generierung eines neuen Sudokus wird jeweils ein zufälliges Sudoku aus dem Korpus ausgewählt.

### Transformieren

Das ausgewählte Sudoku aus dem Korpus wird anschliessend den verschiedenen Transformationen unterzogen. Alle Transformationen sind unter Zuhilfenahme eines Zufallsfaktors realisiert. Genauer gesagt werden die Permutationen für die einzelnen Transformationen zufällig gewählt. Beim Transponieren beschränkt sich der Zufallsfaktor auf die Entscheidung ob transponiert werden soll oder nicht.

Zuerst wird das Sudoku transponiert, danach werden die Zahlengruppen permutiert und zum Schluss werden die Zeilen, Spalten und Blocks vertauscht.

### Ergänzen

Zum Schluss wird das transformierte Sudoku mit weiteren Zahlen ergänzt. Die Anzahl der wird zufällig ausgewählt, so dass die abschliessende Anzahl Startzahlen innerhalb des angegebenen Bereiches ist.

Für dieses Verfahren haben wir drei verschiedene Techniken implementiert:

* Technik 1: Zufällig Zahlen hinzufügen

Alle Zellen die Im gegebenen Sudoku keinen Wert haben in einer Liste sammeln. Diese Liste mischen und anschliessend entsprechend der gewünschten Anzahl minus die 17 Startzahlen Zahlenwerte hinzufügen. Dafür wird das Sudoku gelöst und die Zahl womit die Zelle gefüllt wird, wird aus der Lösung extrahiert.

* Technik 2: Gespiegelte Zahlen hinzufügen

Für jede gegebene Zelle werden die (bis zu 7) gespiegelten Zellen extrahiert und in eine Liste gesammelt. Die Liste wird gemischt und wie bei der ersten Technik die Zahlen aus der Lösung ergänzt.

* Technik 3: Zahlen entfernen

Bei dieser Technik wird das 17er Sudoku zuerst gelöst. Von der Lösung werden anschliessend Zahlen entfernt, bis die gewünschte Anzahl erreicht ist. Dabei wird jeweils eine Zelle aus der Liste aller gesetzten Zellen ausgewählt. Ist die Zelle in der Mitte des Spielfeldes wird si entfernt und eine neue Zelle wird gewählt. Ist die Zelle jedoch nicht in der Mitte so wird mithilfe eines Zufallswertes entschieden ob die Zelle und ihr punktgespiegeltes Gegenstück entfernt wird, oder nur die gewählte Zelle. Nach dem entfernen wird überprüft ob die gewünschte Anzahl erreicht wurde und falls nicht wird eine neue Zelle ausgewählt.

# Resultate

## Lösungsmethoden (gewisse kaum gebraucht)

Bei der Auswertung des Datenpaket 1 (Abbildung 8) haben wir festgestellt, dass es eine klare Grenze zwischen den Schwierigkeitsstufen Medium und Hard existiert. Alle Sudokus, die als Medium oder tiefer eingestuft sind, können mit Naked und Hidden Singles komplett gelöst werden.

Weiterhin ist die Methode Block-Line Interactions im Datenpaket 1 nur von Sudokus der Stufe Evil / Exotic verwendet worden. Da jedoch Subsets der Grösse drei und höher vereinzelt in den Schwierigkeitsstufen Hard und VeryHardExpert verwendet wurden, stellt sich die Frage ob die Eingliederung der Block-Line Interactions zwischen den Subsets der Grösse zwei und drei richtig ist.

## 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.

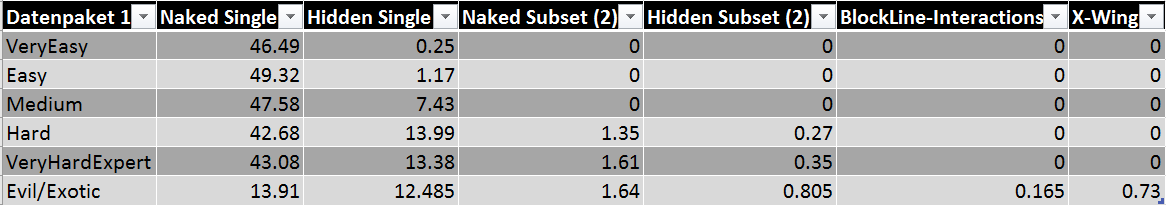


Abb. : Durchschnittliche Anzahl Lösungsmethoden pro Schwierigkeitsstufe

## 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. 9: 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. 10: Konfusionsmatrix basierend auf erstem Datenpaket (Neuroph)

## Generator

Die Resultate des Generators sind stark abhängig von der gewählten Technik um Zahlen zu ergänzen. Für die Tests wurden jeweils 2000 Sudokus generiert und anschliessend mit Neuroph einer Schwierigkeitsstufe zugeteilt. Zusätzlich wurde für jede Technik die Zeit gemessen, die gebraucht wurde um die Sudokus zu generieren.

Technik 1 generierte die 2000 Sudokus innerhalb 1454 Millisekunden. Die Verteilung der Schwierigkeitsstufen ist bei dieser Methode gleichmässig verteilt, wobei die mittleren Schwierigkeiten etwas häufiger generiert werden. Der Nachteil dieser Technik liegt daran, dass die Symmetrie des generierten Sudokus nicht beachtet wird.

Technik 2 brauchte 1514 Millisekunden für die folgenden Resultate. Wie im Diagramm ersichtlich werden bei dieser Technik deutlich mehr Sudokus der Schwierigkeiten Medium und Hard generiert. Wie auch bei Technik 1 kann die Symmetrie der generierten Sudokus auch bei dieser Technik nicht garantiert werden, da die 17 Startzahlen keine Anforderungen an Symmetrie haben. Die Chance, dass ein Sudoku erstellt wird, das gewisse Symmetrien erfüllt ist jedoch höher als bei der ersten Technik.

Technik 3 hat dadurch, dass die Zahlen in symmetrischen Paaren entfernt werden eine hohe Symmetrierate. Es werden jedoch fast ausschliesslich einfache Sudokus generiert. Zudem werden bei der Generierung viele Sudokus verworfen, weil sie unser Programm nicht lösen kann, dies zeigt sich auch in der langen Generierungszeit von 960 Sekunden für die 2000 Sudokus.

## 

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.

# Schluss

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.

# 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.