IP5 - NLP Schweizerdeutsch

Fabio Strappazzon, Matthias Ernst

Inhalt

[Einführung 3](#_Toc499555325)

[Zusammenfassung 3](#_Toc499555326)

[Problemstellung 3](#_Toc499555327)

[Vorgehen 3](#_Toc499555328)

[NLP 4](#_Toc499555329)

[Daten 5](#_Toc499555330)

[Tasks 6](#_Toc499555331)

[Task 1 6](#_Toc499555332)

[Satzbewertung 6](#_Toc499555333)

[Ausfilterung 6](#_Toc499555334)

[Wortalignierung 6](#_Toc499555335)

[Fehlerkorrektur 9](#_Toc499555336)

[Ergebnis 9](#_Toc499555337)

[Task 2 10](#_Toc499555338)

[Ergebnis 11](#_Toc499555339)

[Schlussfolgerung 12](#_Toc499555340)

Titelblatt  
Abstract  
Einleitung  
Hauptteil  
Schlussfolgerung, Fazit

# Einführung

## Zusammenfassung

In dieser Arbeit werden Übersetzungen und Transkriptionen, die auf der Website [www.dindialaekt.ch/](http://www.dindialaekt.ch/) erfasst wurden, mithilfe von NLP-Tools analysiert und verarbeitet.

## Projektkontext

Auf der Website [www.dindialaekt.ch/](http://www.dindialaekt.ch/) können Benutzer unter Anderem zwei Tasks erledigen:

1. Transkription schweizerdeutschen Audiofiles
2. Übersetzung von schweizerdeutschem Text auf Hochdeutsch

Zu jedem Audiofile und jedem schweizerdeutschen Ausgangstext gibt es mehrere (etwa 3 bis 7) zusammengehörende Transkriptionen / Übersetzungen.

Aus den so gewonnenen Transkriptionen und Übersetzungen sollen nun über Wortalignierung mögliche Schreibweisen für schweizerdeutsche Ausdrücke (Task 1) und mögliche schweizerdeutsch Übersetzungen für hochdeutsch Ausdrücke gefunden werden (Task 2).

Die Aufgabe wird dadurch erschwert, dass viele Transkriptionen unvollständig (Mit Platzhaltern in Form von \*\*\* oder ???) oder gar komplett falsch sind. Weiter haben Benutzer bei Task 2 die Freiheit, die Satzstellung frei zu wählen.

## Problemstellung

* Unbrauchbare Transkriptionen ausfiltern (Standard-Deutsch)
* Gute Transkriptionen erkennen (Vollständige)
* \*\*\* Auslassungen interpolieren aus guten Transkriptionen
* Explizite Alternativen behandeln
* Schreibvarianten von Wörtern und Ausdrücken erkennen über String-Alignierung
* Alignierung von schweizerdeutschen Schreibvarianten

Daraus ergeben sich uns folgende zwei Fragestellungen:  
Welche Tools funktionieren mit unseren Daten?  
Welche und wie viele Daten brauchen wir für gute Resultate?

## Vorgehen

Beide Tasks erfordern eine ähnliche Herangehensweise: Erst muss ein Algorithmus gefunden werden, der die vorhandenen zusammengehörigen Übersetzungen und Transkriptionen qualitativ bewertet. Dann muss anhand dieser Bewertung entschieden werden, welche Übersetzungen und Transkriptionen brauchbar und unbrauchbar für das weitere Verfahren sind. Mit den verbleibenden Sätzen können nun mit Satz / Wortalignment Tools Gruppen von Ausdrücken gebildet werden, die bedeutungsgleich sind.

# Daten

Tabellarische Daten. Für swg2g folgende relevante Spalten:

TASK\_ID: identifiziert zusammengehörige Übersetzungen

INFO: vom Benutzer erfasste hochdeutsch Übersetzung

REF: schweizerdeutscher Originaltext

Auszug als CSV:

VALID;TASK\_ID;TASK\_RUN\_ID;USER\_ID;INFO;REF\_ID;REF

True;2530;16287;2887;das war dann bereits zu einem späteren Zeitpunkt, das war als....;BS:0012;das isch den scho schpööter gsii das isch gsii woo

True;2530;19101;4117;Das war bereits später, das war als;BS:0012;das isch den scho schpööter gsii das isch gsii woo

True;2531;453;1777;und mit neunzehn ist man in die RS eingerückt;BS:0016;und mit nüünze isch men in d eress iigruggt

True;2531;2245;1995;und mit 19 ist man in die RS eingerückt.;BS:0016;und mit nüünze isch men in d eress iigruggt

Für Transcribe folgende relevante Spalten:

TASK\_ID: identifiziert zusammengehörige Transkriptionen

INFO: vom Benutzer erfasste schweizerdeutsch Transkription

Auszug als CSV:

URL;VALID;TASK\_ID;TASK\_RUN\_ID;USER\_ID;INFO;REF

AUDIO;True;1829;24287;2887;Ma het dénn alz zäme glääseni Lüt was me hät chöne zämetriibe allz was Chopf u Loch hégi müessi ez höwe héts aube ghéésse;https://dindialaekt.ch/data/transcribe/SDS\_CD1\_1\_11\_speaker1\_1.mp3

AUDIO;True;1829;30820;3563;ma het denn allz zäme glääsenei lüt was mer het chöne zämetribe alts was chopf u loch heegi müesi ez höve hets aube gheesse;https://dindialaekt.ch/data/transcribe/SDS\_CD1\_1\_11\_speaker1\_1.mp3

AUDIO;True;1830;6158;3019;mer s alles us grückt d erste het dri gschnitte es paar schritt gange e zwetei made e dritt, e virtte bi al zäme drin gis sind;https://dindialaekt.ch/data/transcribe/SDS\_CD1\_1\_11\_speaker1\_2.mp3

AUDIO;True;1830;10807;2218;när isch also alles usgrückt dr erschte het dri gschnitte es paar schritt gange zweite id maade e dritte e vierte bis au zäme drin gsi si;https://dindialaekt.ch/data/transcribe/SDS\_CD1\_1\_11\_speaker1\_2.mp3

## Charakteristiken der Daten

TODO

# Tasks

## Task 1

### Satzbewertung Zum Sätze bewerten, muss man sie auf Ähnlichkeit prüfen.

Die Bewertung der Sätze beruht auf der Prämisse, dass «gute» Transkriptionen sich ähneln und «schlechte» Transkriptionen sich von den anderen stark unterscheiden.

So könnte beispielsweise die (gewichtete) Levenshtein-Distanz (Editierdistanz) verwendet werden: Es wird die Levenshtein-Distanz zwischen jedem Satzpaar berechnet. Derjenige Satz, der zu allen anderen die tiefste Levenshtein-Distanz hat, ist folglich der «ähnlichste» und somit beste Satz.

Das gleiche Prinzip ist anwendbar mit Tools für die automatische Bewertung von Maschinenübersetzungen.

Word Error Rate ist eine Metrik die auf dem gleichen Prinzip wie die Levenshtein-Distanz basiert, jedoch auf Wortebene agiert, anstatt auf Phonem-Ebene.

BLEU ist ein Algorithmus, dessen Bewertung sehr gut mit deren von Menschen korrespondiert. Er ist viel verwendet und hat gute Performance. Integriert in NLTK

Einige Metriken, die BLEU-basiert sind, und versuchen dessen Schwächen zu verbessern: TER, METEOR, NIST

Weiter: LEPOR / hLEPOR

Um Outlier zu finden, muss man die Sätze bewerten können.

### Ausfilterung

Methoden zum Ausfiltern:

Statistik: Outlier mithilfe von Standardabweichung o.Ä. erkennen. Problem: Daten sind nicht normalverteilt, zu kleine Datenmenge

Fixer Minimalwert: gute Baseline. Für BLEU sollte dieser Wert etwa zwischen 0.3 und 0.6 liegen.

Clustering: Da es zwingend mehrere gute Sätze geben muss, damit diese als gut erkannt werden, können eindimensionale Clustering Algorithmen verwendet werden und dann die höchsten 1-2 Cluster als akzeptiert angesehen werden. Problem: Wahl des Clustering-Algorithmus in Anbetracht der extrem kleinen Datenmenge.

Iteratives Filtern und Bewerten: Solange sich die durchschnittliche Bewertung deutlich verbessert, wird der schlechteste Satz entfernt und alle anderen Sätze ohne den entfernten neu bewertet. Probleme: was ist eine «deutliche» Verbesserung? Die Tendenz der Daten zu gruppieren verfälscht das Ergebnis weiter

### Wortalignierung

Obwohl es sich hierbei um ein Wortalignierungsproblem handelt, ziehen wir nur Satzalignierer in Betracht. Dies aus folgendem Hintergrund:

Wortalignierer alignieren Wörter aus unterschiedlichen Sprachen, in denen die Wortstellung völlig unterschiedlich sein können. Satzalignierer gehen generell von der gleichen Satzreihenfolge in beiden Texten aus. In unseren Texten wisse wir, dass die Wortreihenfolge immer die selbe sein wird. Es können jedoch einzelne Wörter fehlen oder Wortpaare zusammengeschrieben werden. Damit sind normale Wortalignierer zu flexibel, wohingegen wir alle Anforderungen für ein Satzalignierungsproblem erfüllt haben.

Um Satzaligner für Wortalignierungsprobleme zu verwenden, können Sätze als Abschnitte, Wörter als Sätze und Buchstaben als Wörter angesehen werden.

Satzaligner benutzen eines oder mehrere folgender Informationen: Wörterbuch, Textlänge, Textähnlichkeit. Da es in unserem Fall keine Wörterbücher gibt, können alle Tools, die ein Wörterbuch benötigen, ausgeschlossen werden.

Bleualign ist ein Satzalignierungstool, das versucht, den BLEU Score zwischen den alignierten Sätzen zu maximieren.

Hunalign (wenn ohne Wörterbuch verwendet) aligniert in einer ersten Iteration anhand der Satzlänge. Aus der so gewonnenen Information wird ein Wörterbuch generiert, das in einer zweiten Iteration zur Verbesserung der Alignierung verwendet wird.

Vorgehen:

Aus einer Satzgruppe wählen wir Paare von Sätzen und lassen das Alignierungstool entscheiden, welche Ausdrücke zusammengehören. So erhalten wir Gruppen von Ausdrücken, die gleichbedeutend sein sollten.

#### Bewertung der Alignments

Das Ergebnis dieses Prozesses ist eine Liste von Gruppen von Ausdrücken. Die Ausdrücke einer Gruppe sollten hierbei gleichbedeutend sein. Da wir wissen, dass die Ausdrücke das Resultat von Transkriptionen derselben Audiodatei sind, sollten die Ausdrücke bis auf die Schreibweise identisch sein. Fehlerhafte Transkriptionen, schlechte Alignment-Ergebnisse und fehlerhaftes Zusammenbauen der Ausdrucksgruppen verschlechtern jedoch das Ergebnis. Deshalb müssen wir einen Weg finden, das generierte Alignment qualitativ zu bewerten.

Algorithmen

**Bleu-Score:**

BLEU ist für die Bewertung längerer Texte gedacht und liefert beim Vergleich einzelner Wörtern kein sinnvolles Ergebnis.

**Levenshtein-Distanz**

Auch Editierdistanz genannt, berechnet die Anzahl an Operationen (Einfügen, Löschen, Ersetzen) die es braucht, um von Wort A auf Wort B zu kommen.

Beispiel: Geld und Gäld haben eine Distanz von 1.

**Gewichtete Levenshtein-Distanz**

Besser geeignete Version von der Levenshtein Distanz für das Vergleichen von Phonetisch gleich Klingenden Worten. So kann man hier mit der Gewichtung der Operationen auf den Buchstaben eine fairere Editierdistanz berechnen. Beispiel: Gäld und Geld klingen im Schweizerdeutschen sehr ähnlich, würden aber mit der normalen Version eine Distanz von 1 aufweisen. Jedoch hier bei der Gewichtung wird die Substitution von gewissen Zeichenpaaren billiger gemacht, wenn sie akustisch ähnlich sind.

<https://github.com/nadvornix/python-fizzle>

<https://pypi.python.org/pypi/weighted-levenshtein/0.1>

**Damerau-Levenshtein-Distanz**

Die Operationen der Levenshtein Distanz wird um Transponieren erweitert, diese ermöglicht das Vertauschen von zwei nebeneinanderstehenden Buchstaben.

<https://github.com/gfairchild/pyxDamerauLevenshtein>

**Soundex**

Errechnet die Lautähnlichkeit von zwei Strings. Gedacht um englische Namen zu vergleichen. Beispiel: Smith und Smiff.  
Schlechte Unterstützung für Deutsch, keine für CH-Deutsch.

<https://pypi.python.org/pypi/soundex>

**Metaphone**

Generiert Keys basierend auf Lautähnlichkeit eines Wortes. Im Englischen kann man so zwei Wörter vergleichen. Beispiel: Programming und Programmer resultieren beide in PRKRM.   
Schlechte Unterstützung für Deutsch, keine für CH-Deutsch.

<https://github.com/dracos/double-metaphone>

**Entscheid über den Algorithmus**

Um mal eine Baseline zu schaffen, wählen wir die Gewichtete Levenshtein-Distanz mit kleinen Anpassungen wie zum Beispiel, dass ä und e tiefer bewertet werden. Eine ausführlichere Gewichtung wäre bestimmt noch besser und genauer, jedoch schätzen wir den Aufwand als zu gross ein. Falls man den Algorithmus von Soundex und Metaphone auf Schweizerdeutsch anpassen würde, würden diese bestimmt spannende Ergebnisse liefern, jedoch ist auch dies zu Aufwändig. Zu einem späteren Zeitpunkt kann man sich dies jedoch nochmals überlegen.

Zeitpunkt der Bewertung des Alignments

**Pro Satzpaar**

Zwei Sätze werden aligniert und dann werden die daraus resultierenden Wortpaare bewertet. Schlechte Paare werden in diesem Schritt direkt verworfen, die guten in einem 2d-array abgelegt.

**Pro Satzgruppe**

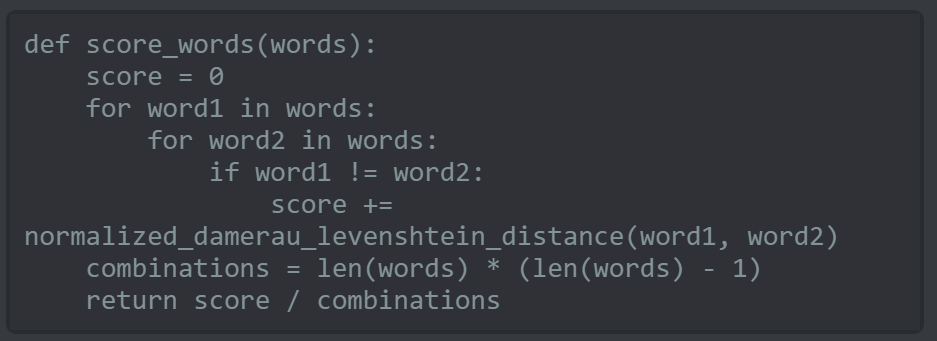
Alle Sätze einer Satzgruppe werden miteinander aligniert, ein 2d-array mit alignierten Wörter wird kreiert. Alle Wörter in diesem Array werden Bewertet, schlechte werden entfernt.

**Global**

Alle Sätze werden miteinander aligniert, ein 2d-array mit alignierten Wörter wird kreiert. Alle Wörter in diesem Array werden Bewertet, schlechte werden entfernt.

**Tester**

Mit folgendem Code kreieren wir eine Bewertung über das 2d-array. Somit hoffen wir den Zeitpunkt, oder auch die Alignierungs-Bewertungsmethode auf Grund von Daten zu optimieren.



**Entscheid**

Müssen zuerst noch Daten sammeln.

### Fehlerkorrektur

### Ergebnis

## Gold Standard für Task 1 Wort Alignierung

Wir haben einen Goldstandard für die Alignierung von Wörtern aus Task 1 erstellt, mit diesem werden wir die verschiedenen Methoden zur Alignierung messen und so vergleichen können.

Sie können diesen in folgendem File  sehen.

Wir haben zum Testen folgende GOLD\_STANDARD\_SET = [2048, 2095, 2080, 2358, 2374, 1842, 1851, 1930, 1934, 1967] Satzgruppen genommen.

## Interpolation von fehlenden Wörtern

bad sentence:

Es isch so das mich sThema interessiert hät aso ich mags nöd weg em Job noch, \*\*\* aso mi intressiert würklich und es isch cool\*\*\*

improved sentence:

Es isch so das mich sThema interessiert hät aso ich mags nöd weg em Job noch, äm- aso mi intressiert würklich und es isch cool\*\*\*

improved sentence(with additional word filter):

Es isch so das mich sThema interessiert hät aso ich mags nöd weg em Job noch, äm- aso mi intressiert würklich und es isch kuul.

## Task 2

# Ergebnis

# Schlussfolgerung