IP5 - NLP Schweizerdeutsch

Fabio Strappazzon, Matthias Ernst



Betreuer: Wolfgang Weck, Daniel Kröni

Auftraggeber: Universität Zürich, Institut für Computerlinguistik

Fachhochschule Nordwestschweiz, Studiengang Informatik

Windisch, 07.01.2018

Diese Arbeit ist im Auftrag des Instituts für Computerlinguistik der Universität Zürich entstanden. Das Institut betreibt das Projekt «Citizen Linguistics: Locate that dialect!», in welchem Benutzer spielerisch schweizerdeutsche Texte klassifizieren, transkribieren und übersetzen.

Das Ziel dieser Arbeit ist die Verarbeitung von schweizerdeutschen Transkriptionen, um unterschiedliche Schreibweisen für dieselben Ausdrücke zu finden. Weiter soll es möglich sein, unvollständige Transkriptionen automatisch zu vervollständigen. Im Gebiet des Natural Language Processing ist dies unter dem Begriff der String-, Wort- oder Textalignierung bekannt. Tools und Algorithmen hierfür existieren zwar, müssen aber auf Verwendbarkeit im Bezug auf unsere Datencharakteristiken untersucht werden. Weiter sind Vor- und Nachbearbeitungsschritte (Ausfilterung unbrauchbarer Transkriptionen, qualitative Bewertung von Transkriptionen und Alignierungen, Kombination unabhängiger Alignierungen) nötig.

Wir stellen ein Verfahren vor, das aufgrund mehrerer Transkriptionen in Form eines CSV-Files eine Liste von Wortgruppen generiert. Wörter und Ausdrücke innerhalb derselben Wortgruppe sind dabei unterschiedliche Schreibweisen mit derselben Bedeutung.

**Inhaltsverzeichnis**

[1 Einführung 4](#_Toc503104620)

[1.1 Zusammenfassung 4](#_Toc503104621)

[1.2 Projektkontext 4](#_Toc503104622)

[1.3 Problemstellung 4](#_Toc503104623)

[1.4 Vorgehen 4](#_Toc503104624)

[2 Daten 6](#_Toc503104625)

[2.1 Charakteristiken der Daten 6](#_Toc503104626)

[3 Alignierung von schweizerdeutschen Transkriptionen 7](#_Toc503104627)

[3.1 Satzbewertung 7](#_Toc503104628)

[3.2 Ausfilterung 7](#_Toc503104629)

[3.3 Wortalignierung 7](#_Toc503104630)

[3.4 Bewertung der Alignments 8](#_Toc503104631)

[3.5 Fehlerkorrektur / Interpolation 9](#_Toc503104632)

[3.6 Bewertung des Ergebnisses 10](#_Toc503104633)

[4 Ergebnis 11](#_Toc503104634)

[5 Schlussfolgerung 12](#_Toc503104635)

# Einführung

## Zusammenfassung

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

## Projektkontext

Im Rahmen des vom Schweizerischen Nationalfods geförderten Agora-Project «Citizen Linguistics: Locate that dialect!» ist die Website [www.dindialaekt.ch/](http://www.dindialaekt.ch/) entstanden.

Auf dieser Website können Benutzer unter Anderem folgenden Task erledigen:

* Transkription schweizerdeutschen Audiofiles
* Identifizieren von Dialekten mittels Lokalisierung auf einer Karte
* Übersetzung von schweizerdeutschen zu hochdeutschen Texten

Im Rahmen dieser Arbeit wollen wir aus den so gewonnenen Transkriptionen weitere Informationen gewinnen.

Zu jedem Audiofile gibt es mehrere (etwa 3 bis 7) zusammengehörende Transkriptionen.

Aus den so gewonnenen Transkriptionen sollen nun über Wortalignierung mögliche unterschiedliche Schreibweisen für schweizerdeutsche Ausdrücke gefunden werden.

Die Aufgabe wird dadurch erschwert, dass viele Transkriptionen unvollständig (Mit Platzhaltern in Form von \*\*\* oder ???) oder gar komplett falsch sind.

## Problemstellung

* Unbrauchbare Transkriptionen ausfiltern
* Gute Transkriptionen erkennen
* \*\*\* Auslassungen interpolieren aus guten Transkriptionen
* Schreibvarianten von Wörtern und Ausdrücken erkennen über String-Alignierung

Daraus ergeben sich uns folgende zwei Fragestellungen:  
**Welche Tools funktionieren mit unseren Daten?**

Natural Language Processing (NLP) ist ein etabliertes Gebiet. Für viele Problemstellungen gibt es Tools und Algorithmen. Viele dieser Tools haben jedoch gewisse Voraussetzungen. Beispielsweise benötigen sie einen Textkorpus einer gewissen Grösse oder sie benötigen Wörterbücher der zu untersuchenden Sprachen. Da es im Schweizerdeutschen keine genormte Rechtschreibung und daher auch keine Wörterbücher gibt, sind uns in der Wahl der Tools einige Grenzen gesetzt. Wir wollen also herausfinden, was für Tools grundsätzlich für unsere Art von Daten verwendbar sind.  
**Welche und wie viele Daten brauchen wir für gute Resultate?**

Die Tools, die wir als «brauchbar» erkennen, werden jedoch auch Grenzen haben. Für die Betreiber der Website ist wichtig zu wissen, dass für eine Audiodatei ein Minimum and Transkriptionen vorhanden sein muss, um sinnvolle Analysen damit anzustellen. Konkret wollen wir also herausfinden, wie viele Transkriptionen zur selben Audiodatei vorhanden sein müssen, damit die von uns angewendeten Techniken funktionieren.

## Vorgehen

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. Folgende relevante Spalten:

TASK\_ID: identifiziert zusammengehörige Transkriptionen

INFO: vom Benutzer erfasste schweizerdeutsch Transkription

Auszug aus 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

Stand 2017-07-08

2848 Transkriptionen zu 554 verschiedenen Audiodateien.

Pro Audiodatei sind zwischen 1 und 14 Transkriptionen vorhanden. Durchschnitt 5.14, Median 5.

So sehen zwei plausible Transkriptionen aus:

«Äs isch die Zyt wo d'Lüt afänd üsszie för d' Härdöpfel understuen fascht alli hän da un dert än Härdöpfelblätz. Ds Grosis ämel o.»

«Es isch die Ziit, wo d'Liit afend üsziehn fer d'Härdöpfu underztüen Faschd alli hend \*\*\* \*\*\* en Häröpfublätz, z'Grosis emu o.»

Die Zweite beinhaltet eine Auslassung (\*\*\*), da der Benutzer den Ausdruck «da un dert» nicht verstanden hat.

So sieht eine unbrauchbare Transkription aus: «dfbdfgh \*\*\* \*\*\* \*\*\* \*\*\* \*\*\*»

# Alignierung von schweizerdeutschen Transkriptionen

## 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. In <http://lium3.univ-lemans.fr/mtmarathon2010/ProjectFinalPresentation/SentenceAlignment/sentence_alignment.pdf> werden einige Satzaligner vorgestellt und verglichen. Die Resultate hierbei sind generell sehr ähnlich, weshalb wir keine fundamental unterschiedlichen Ergebnisse erwarten. Wir haben zwei der vorgestellten Aligner (Bleualign und Hunalign) ausprobiert.

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.

Fastalign ([Chris Dyer](http://www.cs.cmu.edu/%7Ecdyer), [Victor Chahuneau](http://victor.chahuneau.fr), and [Noah A. Smith](http://www.cs.cmu.edu/%7Enasmith). (2013), <http://www.aclweb.org/anthology/N13-1073>) ist ein Wortalignierer. Er bietet jedoch eine Option an, die diagonale Alignierungen bevorzugt. Da sich jedoch das Input- und Outputformat stark von demjenigen von Bleualign und Hunalign unterscheidet und wir davon ausgehen müssen, dass das Resultat nicht fundamental besser ist, haben wir hier Fastalign nicht in Betracht gezogen.

Vorgehen:

Aus einer Satzgruppe wählen wir Paare von Sätzen und lassen das Alignierungstool entscheiden, welche Ausdrücke zusammengehören. So erhalten wir Paare von Ausdrücken, die gleichbedeutend sein sollten. Nun wollen wir diese Ausdruckspaare zu ganzen Gruppen gleichbedeutender Ausdrücke vereinen. Wir sehen also jeden Ausdruck als einen Knoten in einem Graph an. Das Alignierungstool sagt uns, zwischen welchen Knoten eine Kante existiert. Mit jedem Alignment eines Satzpaars werden dem Graph also 0-n neue Knoten und n neue Kanten hinzugefügt. Am Ende dieses Prozesses ist jeder verbundene Teilgraph eine Gruppe gleichbedeutender Ausdrücke.

## Filterung des 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, um dann ungenügende Ergebnisse auszufiltern.

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.

**Double Metaphone**

Generiert Keys auf Lautähnlichkeit eines Wortes wie Metaphone, unterstützt jedoch auch Slawisch, Germanisch, Celtic, Griechisch, Französisch, Italienisch, Spanisch, Chinese und andere Herkünfte.

<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. Als Unterstützung verwenden wir aber noch Double-Metaphone. Durch die Kombination von zwei verschiedenen Algorithmen kommen wir auf bessere Ergebnisse.   
Unsere Algorithmen filtern wie folgt die Alignments aus:

|  |  |  |
| --- | --- | --- |
| Metaphone Bewertung | Levensthein Bewertung | Resultat |
| OK | OK | NICHT AUSGEFILTERT |
| OK | FAILED | NICHT AUSGEFILTERT |
| FAILED | OK | NICHT AUSGEFILTERT |
| FAILED | FAILED | AUSGEFILTERT |

**Hier ein paar Beispiele:**

**Params: Levensthein Filter (0.4)**

|  |  |
| --- | --- |
| Satzpaar mit Levenstheinvalue und Metaphone Bewertung | Wird ausgefiltert |
| ('sägäslì', 'sägessli', 0.275, False) | Nein, da Levensthein OK |
| ('de', 'die', 0.3333333333333333, True) | Nein, da beide OK |
| ('wèrdët', 'wäärded', 0.42857142857142855, True) | Nein, da Metaphone OK |
| ('ëm', 'am', 0.1, True) | Nein, da beide OK |
| ('?ù', 'scho', 1.0, False) | Ja, da beide FAILED |
| ('ùn', 'und', 0.6666666666666666, False) | Ja, da beide FAILED |

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, siehe Algorithmen oben, die guten in einem Graphen abgelegt. Die Wörter sind die Nodes.

**Pro Satzgruppe**

Alle Sätze einer Satzgruppe werden miteinander aligniert, ein Graph mit alignierten Wörter wird kreiert. Alle Wörter in diesem Graphen werden Bewertet, schlechte werden nicht hinzugefügt.

**Global**

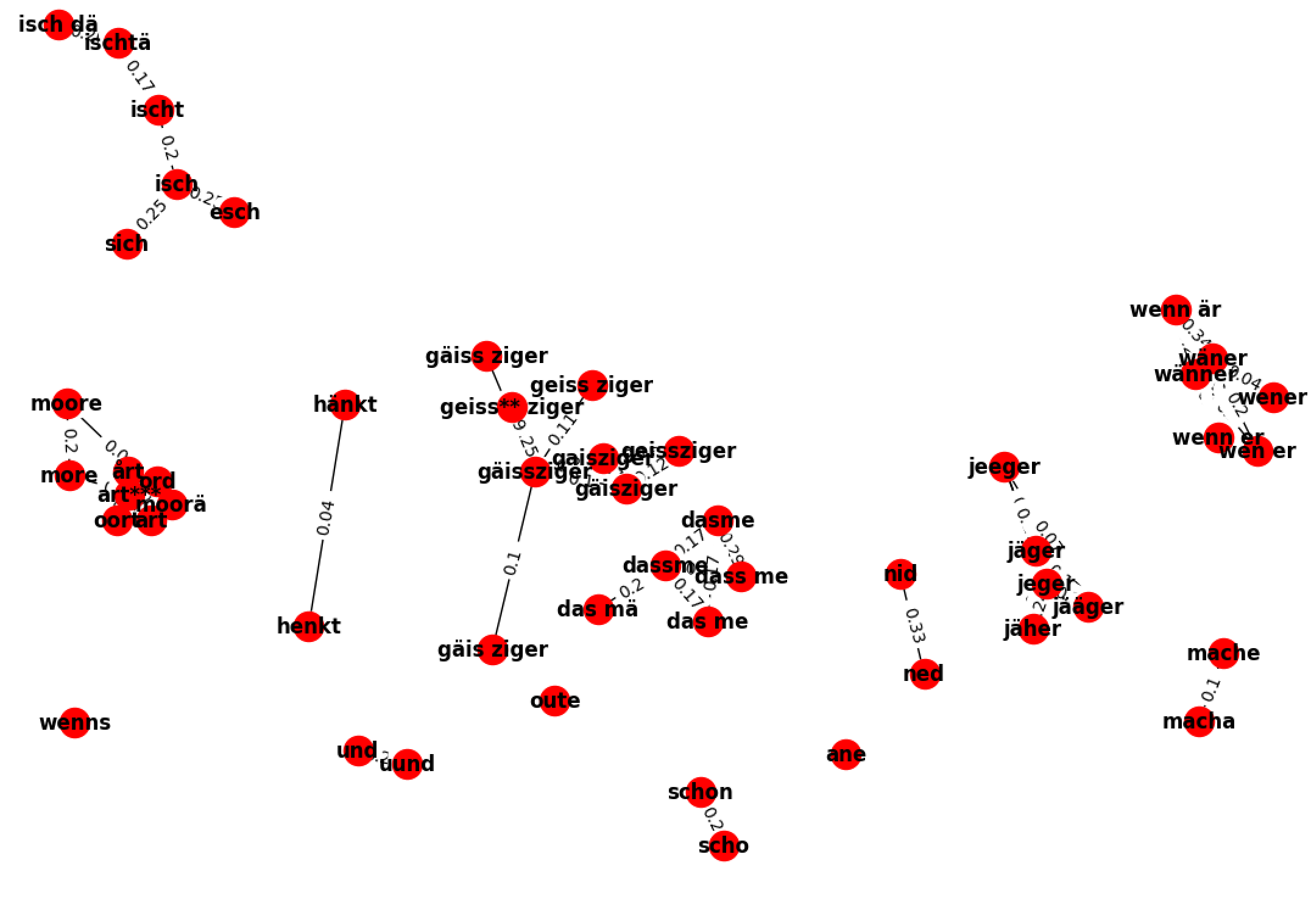
Alle Sätze werden miteinander aligniert, ein Graphen mit alignierten Wörter wird kreiert. Alle Wörter werden vor dem Einfügen in den Graphen bewertet und ausgefiltert.

**Tester**

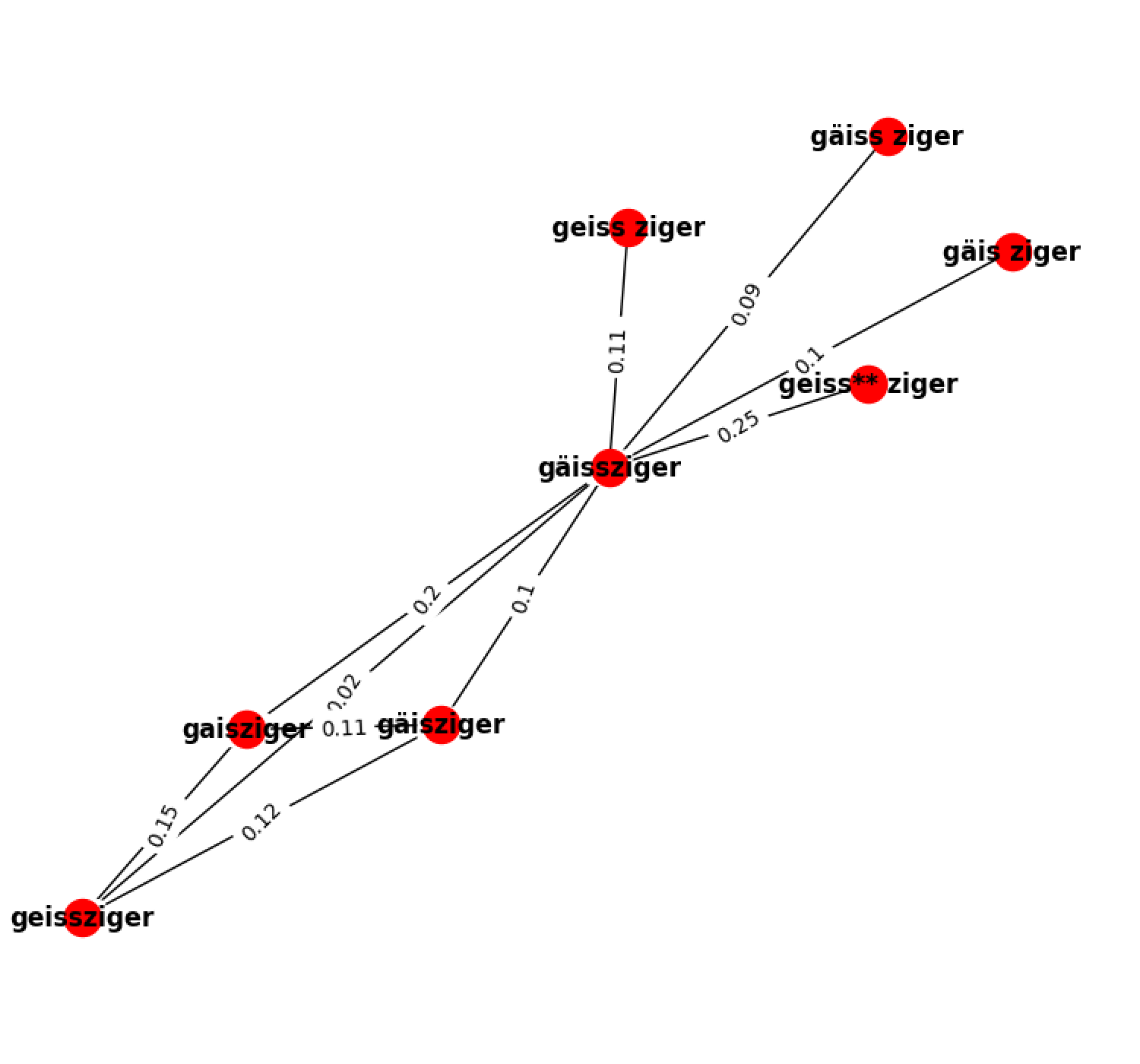
**TODO**

**Entscheid über Parameter des Filters**

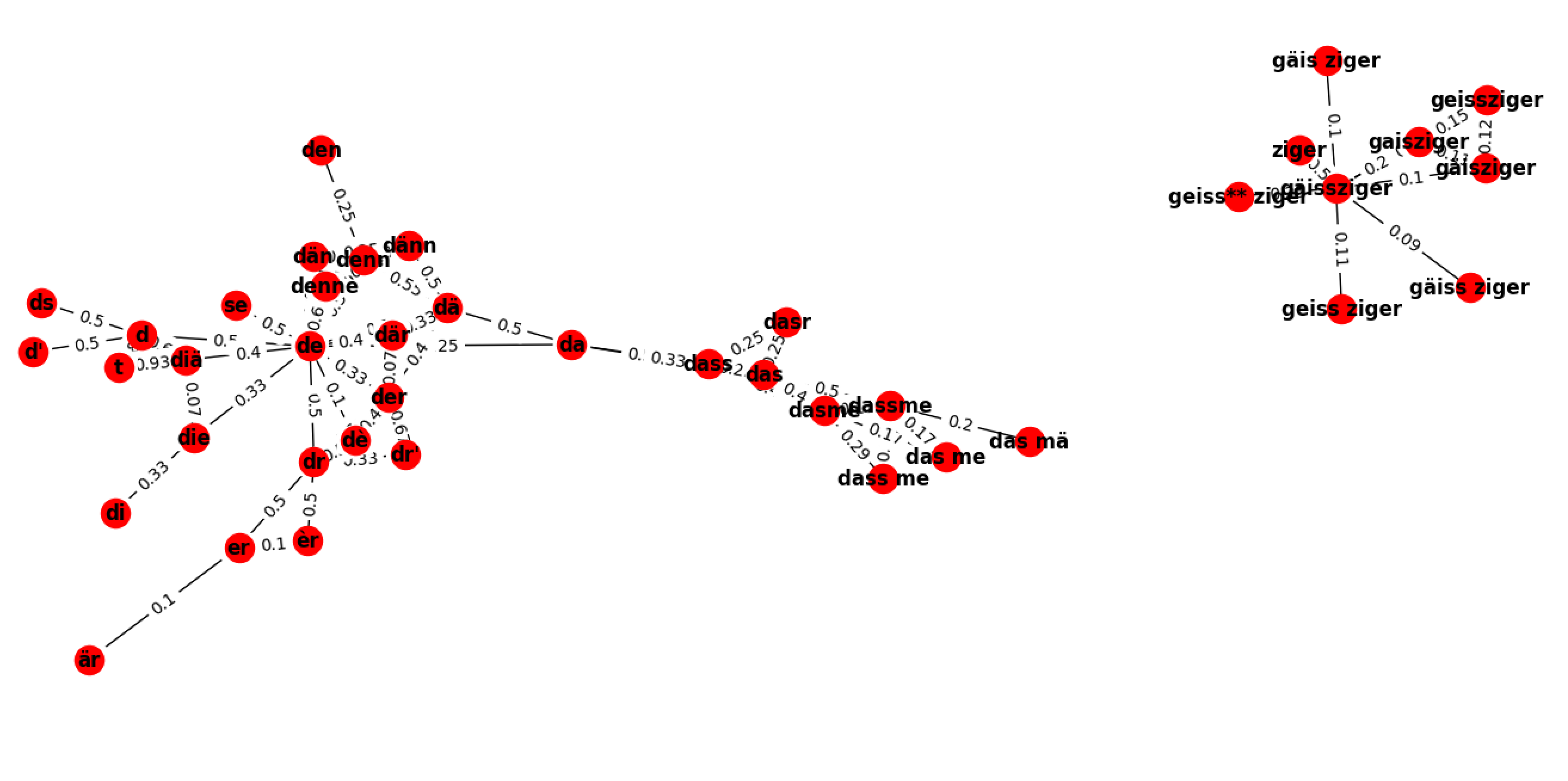
**FIltervalue von 0.25**



**Filtervalue von 0.333**



**FIltervalue von 0.6**



## Fehlerkorrektur / Interpolation

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.

## Bewertung des Ergebnisses

Um verschiedene Parameter und Tools miteinander vergleichen zu können, muss es einen Weg geben, das generierte Alignment qualitativ zu bewerten.

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.

.

Wir haben zum Testen folgende GOLD\_STANDARD\_SET = [2048, 2095, 2374, 1930, 1929] Satzgruppen genommen.

# Ergebnis

# Schlussfolgerung