Universiteit Utrecht

Faculteit Geesteswetenschappen

De invloed van POS-tags op het herkennen van code-switching in Nederlandse tweets

Bachelor Scriptie
7.5 ECTS

Eline Kirsten Kempkes

Kunstmatige Intelligentie

Begeleider:
Dr. F.W. Adriaans

Tweede lezer:
Dr. M.P. Schraagen

Januari 2021

Samenvatting

Code-switching zorgt ervoor dat toepassingen zoals taalherkenning en automatische vertalingen minder accuraat worden. Daarom is het van belang dat er onderzoek wordt gedaan naar classificatiemodellen die code-switching automatisch kunnen herkennen. Dongen (2017) heeft al verschillende classificatiemodellen ontworpen om code-switching te herkennen in Nederlands-Engelse tweets. Het doel van deze scriptie is het verbeteren van deze modellen door het toevoegen van een POS-feature. Uit andere onderzoeken is namelijk gebleken dat dit een positieve invloed kan hebben op de resultaten van de classificatiemodellen. Om dit doel te bereiken zijn eerst POS-tags toegevoegd aan de dataset. Dit is gedaan door gebruik te maken van een combinatie van twee verschillende taggers, een Engelse en een Nederlandse. Vervolgens zijn er drie modellen getraind op de dataset: een Multinomial Naive Bayes model, een Decision Tree Model en een Support Vector Machine model. De modellen zijn getraind op verschillende combinaties van features. Hierbij is telkens ook de POS-feature toegevoegd. Vervolgens zijn de precision, recall en F1-score bepaald. Hieruit bleek dat de POS-feature in alle gevallen de F1-score verbeterde. Echter, het beste model uit deze scriptie is niet beter dan de huidige modellen van Dongen (2017). Dit kan worden verklaard door een verschil in methode en door de willekeur van 5-fold-cross-validation. De conclusie is dat het toevoegen van een POS-feature de classificatiemodellen voor het herkennen van code-switching in Nederlands-Engelse tweets verbetert.

Inhoudsopgave

1 Inleiding

Code-switching wordt de laatse jaren steeds belangrijker in de Kunstmatige Intelligentie (KI) (Çetinoğlu, Schulz en Vu 2016). Code-switching vindt plaats als er meerdere talen door elkaar worden gebruikt, bijvoorbeeld wanneer er in een Nederlandse zin Engelse woorden staan. De volgende tweet is een voorbeeld waarin meerdere code-switches plaatsvinden:

Basic kan dus heel stylish zijn

Code-switching komt met name voor in informele taal (Dongen 2017). Hierdoor vindt het vaak plaats in gesproken taal en op sociale media (Kent en Claeser 2019; Nguyen en Cornips 2016; Solorio, Blair e.a. 2014). Aangezien sociale media steeds meer gebruikt worden, komt code-switching dus ook steeds vaker voor in geschreven teksten (Rijhwani e.a. 2017; Solorio en Liu 2008).

Code-switching wordt al langer onderzocht vanuit sociologisch en taalkundig perspectief en wordt sinds kort ook steeds belangrijker binnen de KI. Vooral in het domein van Natural Language Processing (NLP) wordt er steeds meer onderzoek gedaan naar de problemen rondom code-switching (Çetinoğlu, Schulz en Vu 2016; Solorio, Blair e.a. 2014). De meeste toepassingen van NLP zijn namelijk gemaakt voor eentalige teksten die grammaticaal correct zijn (I. A. Bhat e.a. 2018). Dit verklaart waarom teksten van sociale media niet goed werken voor de huidige toepassingen, aangezien er zowel grammaticale fouten als code-switching aanwezig zijn (I. A. Bhat e.a. 2018; Çetinoğlu, Schulz en Vu 2016; Kent en Claeser 2019; Owoputi e.a. 2013). Zo wordt de techniek van taalherkenning veel minder accuraat wanneer er in een tekst meerdere talen staan (Çetinoğlu, Schulz en Vu 2016; Kent en Claeser 2019). Ook andere technieken verslechteren, zoals parsing, wat een belangrijke techniek is binnen de KI (I. A. Bhat e.a. 2018; Çetinoğlu, Schulz en Vu 2016; Solorio, Blair e.a. 2014). Technieken voor automatische vertalingen en spraakherkenning worden ook minder accuraat wanneer er sprake is van code-switching (Kent en Claeser 2019; Solorio, Blair e.a. 2014). Om deze redenen is het dus belangrijk dat code-switching wordt onderzocht, om zo de technieken te verbeteren.

Om bij te dragen aan het verbeteren van deze technieken zal in dit onderzoek de focus liggen op het herkennen van code-switching in tweets. Dit wordt gedaan door middel van machine learning. Bij machine learning leren classificatiemodellen verschillende patronen herkennen in data. Bij deze modellen worden features gebruikt, bepaalde kenmerken van de data waarvan de computer gebruik maakt. Door deze techniek kan het herkennen van code-switching in tweets mogelijk automatisch worden gedaan. De keuze om voor tweets te kiezen ligt voor de hand. Tweets worden namelijk vaak gebruikt om code-switching te bestuderen, aangezien het er veel in voorkomt (Nguyen en Cornips 2016; Solorio, Blair e.a. 2014). Daarnaast is code-switching een belangrijke reden dat tweets verkeerd worden vertaald (Kent en Claeser 2019), dus het is nodig om dit probleem op te lossen. Zo zijn er al verschillende onderzoeken gedaan naar onder andere Turks-Duitse tweets (Çetinoğlu 2016) en Engels-Spaanse tweets (Xia 2016). Echter, er is nog weinig onderzoek gedaan naar code-switching in Nederlands-Engelse tweets. Om deze reden worden in deze scriptie modellen voor het herkennen van code-switching in Nederlands-Engelse tweets onderzocht.

Daarnaast is er een uitdaging bij deze twee talen: Nederlands en Engels vertonen veel overeenkomsten, waardoor sommige tokens niet uniek zijn voor een bepaalde taal. Een voorbeeld hiervan is het woord *school*, dat zowel in het Engels als in het Nederlands voorkomt met dezelfde spellingswijze (Claeser, Felske en Kent 2017). Ook Engelse leenwoorden in het Nederlands zijn problematisch voor het herkennen van code-switching (Kent en Claeser 2019).

Op het gebied van code-switching in Nederlandse tweets is dus nog niet veel literatuur. Kent en Claeser (2019) hebben een classificatiemodel gemaakt voor de taal van de woorden in Nederlands-Engelse tweets. Om woorden te classificeren maakten ze gebruik van de Engelse en Nederlandse versie van Wikipedia. Dongen (2017) heeft ook een belangrijke bijdrage geleverd. Zij heeft namelijk verschillende taalherkenningsmodellen gemaakt door middel van machine learning voor Nederlandse tweets met code-switching. Dongen (2017) vermoedde dat toevoegen van POS-tags aan de tweets tot betere prestaties zou kunnen leiden van de modellen. POS-tags is een afkorting voor Part Of Speech tags. Het houdt in dat elk woord uit een tweet een label krijgt met zijn woordsoort. Zo wordt er dus bijgehouden of een woord bijvoorbeeld een zelfstandig naamwoord is of een bijvoegelijk naamwoord.

Er zijn al verschillende onderzoeken geweest naar het nut van POS-tags voor het herkennen van code-switching. Verschilllende onderzoeken lieten zien dat het toevoegen van POS-tags leidt tot een verbetering van de resultaten van de modellen (Adel, Vu, Kirchhoff e.a. 2015; Adel, Vu, Kraus e.a. 2013; Attia e.a. 2019; Solorio en Liu 2008; Soto, Cestero en Hirschberg 2018). Dit komt door het feit dat er vooral bij zelfstandige naamwoorden en bijvoegelijke naamworden worden gewisseld van taal (Joshi 1982; Soto, Cestero en Hirschberg 2018). Door Adel, Vu, Kirchhoff e.a. (2015) wordt zelfs gesuggereerd dat POS-tags belangrijker zijn voor het herkennen van code-switching dan de woorden zelf. Deze onderzoeken zijn nog niet uitgevoerd op het gebied van Nederlands-Engelse tweets, maar het vermoeden is dus dat ook bij deze modellen het toevoegen van POS-tags positieve resultaten geeft. Om dit te onderzoeken wordt in dit onderzoek aan de bestaande modellen van Dongen (2017) een POS-feature toegevoegd, om zo te kijken of het de resultaten verbetert.

1.1 Doelen

Het doel van deze scriptie is het verbeteren van classificatiemodellen voor het herkennen van code-switching van Nederlands-Engelse tweets. Het onderzoek zal hierbij voortbouwen op het eerdere werk van Dongen (2017), waarbij al verschillende modellen voor deze taak zijn ontworpen. De onderzoeksvraag luidt: "*Hoe kunnen de classificatiemodellen voor het herkennen van code-switching in Nederlands-Engelse tweets worden verbeterd?*". Concreet wordt dit gedaan door het toevoegen van een POS-feature. De deelvraag die hierbij hoort is "*Wat is de invloed van het toevoegen van POS-tags op de F1-score van de classificatiemodellen?*". Hierbij is de F1-score een maat om de prestatie van het model te meten. Er zal gebruik worden gemaakt van drie verschillende classificatiemodellen, namelijk het Multinomial Naive Bayes model, het Support Vector Machine model en het Decision Tree model. Er is voor deze drie modellen gekozen omdat zij vaak gebruikt worden in deze context, en tot goede resultaten leiden (Barman e.a. 2014; Eskander e.a. 2014; El-Haj, Rayson en Aboelezz 2018; Kent en Claeser 2019; Solorio en Liu 2008). Uiteindelijk zal er worden gekeken of de resultaten worden verbeterd wanneer de modellen gebruik maken van POS-tags, om zo een antwoord te geven op de hoofdvraag. Ten slotte zal het onderzoek worden vergeleken met de resultaten van Dongen (2017), om de verschillen en overeenkomsten te analyseren.

1.2 Leeswijzer

In sectie 2 zal eerst de noodzakelijke theoretische achtergrond worden behandeld. Hierin zullen verschillende concepten worden toegelicht. Daarnaast wordt ook het belang van POS-tags in code-switching beargumenteerd (2.4). Vervolgens zal er in de methode (3) een beschrijving worden gegeven hoe de dataset eruit ziet (3.1). Ook zal er worden besproken hoe de POS-tags worden toegevoegd (3.2). Ten slotte zal worden beschreven hoe de classificatiesystemen in elkaar zitten (3.3). In sectie 4 zullen de resultaten van het onderzoek worden gerapporteerd. Vervolgens worden deze in sectie 5 besproken en vergeleken met het onderzoek van Dongen (2017). Daarnaast worden tekortkomingen van dit onderzoek en suggesties voor vervolgonderzoek besproken. In de conclusie (6) zal er een antwoord worden gegeven op de hoofdvraag.

2 Theoretische achtergrond

In deze sectie zullen verschillende belangrijke concepten worden toegelicht, zoals code-switching en classificatiemodellen. Daarnaast zal ook eerder onderzoek naar dit gebied worden besproken. Ten eerste zal het concept code-switching worden toegelicht in sectie 2.1. Vervolgens worden de verschillende classificatiemodellen en hun werking besproken. Hierbij wordt ook beargumenteerd waarom er voor deze modellen is gekozen. Daarna zal het concept POS-tagging worden uitgelegd. Ten slotte zal er worden beargumenteerd waarom een POS-feature belangrijk is binnen code-switching in sectie 2.4.

2.1 Code-switching

Zoals in de inleiding (1) is aangegeven, vindt code-switching plaats als er in een tekst verschillende talen worden gebruikt. Daarnaast komt het voornamelijk voor in informele taal. Voorheen ging dit vooral om spreektaal, maar door de opkomst van sociale media is er vaker geschreven taal met code-switching. Een belangrijk voorbeeld hiervan zijn tweets, die in dit onderzoek worden onderzocht (Rijhwani e.a. 2017; Solorio en Liu 2008).
In de literatuur wordt soms onderscheid gemaakt tussen code-switching en code-mixing. Er is nog geen consensus bereikt over het exacte verschil tussen deze twee begrippen (Mabule 2015). Code-switching wordt namelijk soms gedefinieerd als het wisselen van talen op zinsniveau (Bokamba 1989; Claros en Isharianty 2009). Het verschil met code-mixing is dan dat bij code-mixing er binnen één zin wordt gewisseld van taal (Claros en Isharianty 2009). Echter, code-switching wordt soms ook onderverdeeld in intrasentential code-switching, waarmee ook het wisselen binnen zinnen wordt bedoeld (Poplack 2000). Daarnaast worden door sommige onderzoekers de termen door elkaar heen gebruikt (Mabule 2015).

Omdat er dus nog geen consensus is bereikt over het verschil tussen deze begrippen, zal er in deze scriptie alleen de term code-switching worden gebruikt. Hierbij wordt wel onderscheid gemaakt tussen de verschillende niveaus van code-switching. Deze worden intersentential, intrasentential en intramorfeem genoemd. Dit is respectievelijk het wisselen van talen tussen zinnen, binnen zinnen en binnen woorden is. Een voorbeeld van code-switching op intramorfeem niveau is het woord '*restarten*', waarbij Engels en Nederlands door elkaar heen worden gebruikt. Aangezien in dit onderzoek tweets worden bestudeerd, en deze niet vaak uit meerdere zinnen bestaan, zal de focus liggen op intrasentential code-switching. Intramorfeem code-switching komt ook voor in de dataset, maar zo weinig dat het verwaarloosbaar is (zie 3.1).

2.2 Classificatiemodellen

In dit onderzoek zullen er drie verschillende classificatiemodellen gebruikt. Hiermee zal code-switching worden herkend in Nederlands-Engelse tweets. Verschillende studies hebben met een soortgelijke aanpak bij andere talen succesvolle resultaten behaald Adel, Vu, Kraus e.a. (2013), Jaech e.a. (2016), Jain en R. A. Bhat (2014) en Samih e.a. (2016). Classificatiemodellen zijn in feite wiskundige functies die waarden afbeelden op een bepaalde klasse (Abu Mostafa, Magdon-Ismail en H.-T. Lin 2012). In context van code-switching zijn deze waarden, oftewel features, bepaalde kenmerken van woorden. Voorbeelden hiervan zijn de lengte of taal van het woord. De klasse waarop de features worden afgebeeld is in deze context binair. De klasse heeft als waarde *"wel code-switch"* of *"geen code-switch"*. Bij machine learning worden deze features bij elkaar gebundeld een vector. Elk woord in een tweet heeft zijn eigen vector. Bij elke vector, en dus bij elk woord, hoort een waarde van de klasse: -1 bij geen code-switch en 1 bij wel code-switch. Het doel is dat een model bij een bepaalde vector de waarde van de klasse kan voorspellen. Hiervoor wordt het model eerst getraind op een bepaald deel van de dataset. In dit onderzoek is er sprake van supervised learning, dat houdt in dat de waarden van de klassen bekend zijn voor het model (Abu Mostafa, Magdon-Ismail en H.-T. Lin 2012). Hierdoor kan het model bepaalde patronen ontdekken tussen de features en de klassen. Het idee is dan dat het model bepaalde patronen leert en zo ook bij nieuwe data (nieuwe vectoren) de juiste klasse kan voorspellen. In deze context wordt er dus een model getraind op een bepaalde set tweets. Hierbij is aangegeven of er een code-switch plaats vindt. Vervolgens moet dit model bij een nieuwe set tweets aangeven of er code-switches zijn, gebruikmakend van de patronen uit de trainingsdata.

Er zijn verschillende classificatiemodellen die deze taak kunnen uitvoeren. Ze verschillen in hoe zij de data gebruiken om patronen te herkennen. In deze scriptie zullen dezelfde modellen worden gebruikt als in het

onderzoek van Dongen (2017), aangezien het deels wordt nagebootst. Deze modellen zijn het Multinomial Naive Bayes model, het Decision Tree model en het Support Vector Machine model. Al deze drie modellen zijn eerder in de context van code-switching gebruikt (Barman e.a. 2014; Eskander e.a. 2014; El-Haj, Rayson en Aboelezz 2018; Kent en Claeser 2019; Solorio en Liu 2008). Hierbij volgt een overzicht van de verschillende modellen die worden gebruikt in dit onderzoek evenals de argumenten om deze modellen te gebruiken.

2.2.1 Multinomial Naive Bayes model

Een veelgebruikt model bij het herkennen van code-switching is het Naive Bayes model (NB model). Hierbij geldt de aanname dat alle features onafhankelijk van elkaar zijn. Het model wordt vaak gebruikt omdat snel en het makkelijk implementeerbaar is. Ondanks de simpliciteit kan het tot goede resultaten leiden, zoals bij Zampieri e.a. (2014). Bovendien beweert Mahata e.a. (2020) dat zelfs met een kleine dataset NB tot goede resultaten kan leiden. In andere onderzoeken wordt ook veelvuldig gebruik gemaakt van het NB model (El-Haj, Rayson en Aboelezz 2018; Mahata e.a. 2020; Solorio en Liu 2008). In Solorio en Liu (2008) zijn de resultaten zelfs hoger dan bij het meer geavanceerdere Support vector machine model. In dit onderzoek wordt gebruikt gemaakt van een Multinomial Naive Bayes classificatiemodel (MNB), wat een implementatie van NB is. Deze variatie wordt vaker gebruikt voor het classifieren van teksten, omdat het multinomiale gedistribueerde data kan verwerken (Dongen 2017).

2.2.2 Decision Tree model

Bij een Decision tree model worden de verschillende klassen gerepresenteerd door de *leaves*. In dit onderzoek zijn er twee verschillende klassen mogelijk: wel een code-switch, of geen code-switch. De leaves zijn uiteindelijk de uitkomsten van het model, dus de voorspelde klasse. De weg ernaartoe wordt bepaald door de vertakkingen: punten waarop de data wordt gesplitst. De combinaties van features bepalen wanneer er wordt afgesplitst en welke keuze er wordt gemaakt. Uiteindelijk komt de vector bij een leave terecht, dan wordt er voorspeld of er een code-switch plaatsvindt.

Het Decision tree model is al vaker gebruikt in de context van code-switching (Çetinoğlu, Schulz en Vu 2016; Dongen 2017; El-Haj, Rayson en Aboelezz 2018; Kent en Claeser 2019). In Eskander e.a. (2014) is laten zien dat het Decision Tree model in sommige gevallen beter leek te werken dan het Support Vector Machine model. Ten slotte had in de scriptie van Dongen (2017) het Decision Tree model de beste resultaten. Het is dus interessant om te kijken of dit model verbeterd kan worden door POS-tags.

2.2.3 Support Vector Machine model

Een Support Vector Machine (SVM) is een lineair classificatiemodel waarbij gebruik gemaakt wordt van lineaire regressie. Er wordt gebruik gemaakt van een soft-margin SVM, waardoor het model kan werken met data die niet lineair separabel is. SVM wordt onder andere gebruikt bij (Das en Gambäck 2015; Eskander e.a. 2014; El-Haj, Rayson en Aboelezz 2018; Kim en Park 2007). Bij Barman e.a. (2014) werd het SVM model vergeleken met onder andere een k-nearest neighbour model, waarbij het SVM model de beste resultaten had. Het SVM model wordt dus vaker in deze context gebruikt, daarom is het interessant om het ook toe te passen op Nederlands-Engelse tweets.

2.3 POS-tagging

Het doel van dit onderzoek is om te bepalen of het toevoegen van POS-tags als feature de resultaten van de classificatiemodellen verbetert. In deze sectie zullen de POS-tags en POS-taggers nader worden toegelicht. Bij POS-tagging wordt bij elk woord in een zin aangegeven welke woordsoort het is. Zo wordt bijgehouden of het zelfstandige naamwoorden of werkwoorden zijn. Binnen de taalkunde is er veel debat over wat precies de woordsoorten zijn en hoeveel er zijn Hopper en Thompson (1984). Voor deze scriptie is er een onderscheid gemaakt tussen 11 verschillende categorieën: werkwoorden (V), zelfstandige naamwoorden (N), voornaamwoorden (Pron), bijvoegelijke naamwoorden (Adj), bijwoorden (Adv), voorzetzels (Prep), voegwoorden (Conj), lidwoorden (Art), telwoorden (Num), interjecties (Int) en overig (Misc). Een voorbeeldzin laat zien hoe elk woord een eigen POS-tag heeft:

De	tekst	van	het	arrest	is	nog	niet	beschikbaar
Art	N	Prep	Art	N	V	Adv	Adv	Adj

Er is al veel onderzoek gedaan naar het ontwerpen van POS-taggers, modellen die aan elk woord een POS-tag toeschrijven (Brants 2000; Cutting e.a. 1992). In deze scriptie wordt voor het taggen van Nederlandse woorden een eigen tagger gemaakt. Hierbij wordt gebruik gemaakt van combinatie van een unigramtagger en een default-tagger. Een unigramtagger schrijft aan elk woord de tag toe die het meeste voorkomt. Het is namelijk mogelijk dat een woord twee verschillende tags kan hebben, zo kan het woord '*fluit*' zowel een werkwoord zijn (ik fluit), als een zelfstandig naamwoord (een fluit). De unigramtagger kiest dan voor de tag die het meeste voorkomt bij dit woord in de trainingsdata.

In dit onderzoek wordt de unigramtagger gecombineerd met een defaulttagger. Wanneer de unigramtagger faalt, dan wordt er gebruikt gemaakt van een defaulttagger. Een defaulttagger kent simpelweg dezelfde tag aan elk woord toe: dus bijvoorbeeld dat elk woord een zelfstandig naamwoord is. Er is voor deze aanpak gekozen omdat deze combinatie van taggers leidt tot betere resultaten dan wanneer een enkele tagger wordt gebruikt.

2.4 Code-switching en POS-tags

In deze sectie zal worden beargumenteerd waarom een POS-feature kan leiden tot betere resultaten bij het herkennen van code-switching. Dit wordt al laten zien in een van de eerste onderzoeken naar code-switching, uitgevoerd door Solorio en Liu (2008). In dit onderzoek worden code-switching punten voorspeld door een NB model in een Engels-Spaanse tekst. Uit de resultaten blijkt dat de POS-feature in het model leidt tot een verbetering.

Een soortgelijke bevinding komt van Adel, Vu, Kraus e.a. (2013). In deze studie werd geprobeerd code-switching te voorspelen in een Mandarijn-Engelse tekst. Het model was een recurrent neuraal network, waarbij ook POS-tags een rol speelden. Ook hieruit bleek dat deze feature een belangrijke rol speelt bij het herkennen van code-switching. De verklaring hiervoor is dat er vooral bij zelfstandige naamwoorden van taal gewisseld werd.

Ook het onderzoek van Soto, Cestero en Hirschberg (2018) laat zien dat er een sterke relatie is tussen code-switching en de POS-tag van woorden in het Engels-Spaans. Hieruit bleek dat vooral bij bijvoegelijke naamwoorden, zelfstandige naamwoorden en onderschikkende voegwoorden wordt gewisseld van taal. Dit sluit aan bij de theorie van Joshi (1982): voornamelijk woordsoorten uit open klassen worden gebruikt bij code-switching.

Ten slotte laat ook het onderzoek van Attia e.a. (2019) zien dat POS-tags van belang zijn bij code-switching. Hierbij werd code-switching herkend tussen verschillende dialecten van het Arabisch. Hierdoor lieten zij zien dat een POS-tag een sterke aanwijzing is om code-switching te voorspelen. Daarnaast beweert Adel, Vu, Kirchhoff e.a. (2015) in zijn onderzoek dat POS-tags betrouwbaarder zijn om code-switching te herkennen dan de woorden zelf.

In deze studie zal worden onderzocht of dit ook voor Nederlands-Engelse tweets geldt. Uit de dataset die wordt gebruikt blijkt dat de meeste Engelse woorden in de Nederlandse tweets zelfstandige naamwoorden zijn (38.6%). Hierna volgen bijvoeglijke naamwoorden met 15.4% en werkwoorden met 14% (Dongen 2017). Dit is in lijn met de eerdere bevindingen, zoals van Soto, Cestero en Hirschberg (2018). Om deze reden is het interessant om te onderzoeken of het toevoegen van POS-tags als feature ook de resultaten van de Nederlands-Engelse modellen verbeteren.

3 Methode

In deze sectie zal de methode worden behandeld. Allereerst zal de dataset worden besproken. Vervolgens wordt toegelicht hoe er POS-tags zijn toegevoegd aan de data. Uiteindelijk zal het opstellen van de classificatiemodellen worden uiteengezet.

3.1 Dataset

De dataset die in dit onderzoek is gebruikt is gemaakt door Dongen (2017)[1]. De dataset bestaat uit 1300 verschillende tweets, die zijn verkregen door gebruik te maken van de Twitter Search API. Deze tweets hebben door de API het label 'nl' gekregen, maar kunnen wel woorden uit andere talen bevatten. Deze 1300 tweets zijn handmatig geannoteerd, waarbij elk token (19464 in totaal) een taallabel heeft gekregen. De zes verschillende labels zijn: Nederlands, Engels, Gemixt, Sociale Media Term, Andere taal en Onduidelijk. Bij gemixt is er sprake van intramorfeme code-switching (zie 2.1). Onder Sociale Media Termen vallen bijvoorbeeld gebruikersnamen, hashtags en emoticons. Het label Andere taal houdt in dat het woord niet Nederlands en niet Engels is. De tag Onduidelijk is gegeven aan woorden die niet in een van de andere categorieën vielen.

Van alle 19464 verschillende tokens vielen de meeste in de categorie Nederlands, namelijk 85.8%. Daaropvolgend zijn Sociale Media Termen (11.5%) en Engels(1.4%). Engels is dus slechts een klein deel van de dataset. Echter, 8.5% van de tweets bevat Engelse tokens, dus in een redelijk deel is code-switching aanwezig. Slechts 0.1% van de tokens heeft het label Gemixt. Om deze reden wordt intramorfeme code-switching buiten beschouwing gelaten.

Ten slotte is de betrouwbaarheid van deze annotatie getest met behulp van een inter-annotator agreement door Dongen (2017). Hierbij is de waarde van Cohen's kappa 0.94, wat betekent dat de annotatie betrouwbaar is.

3.2 Toevoegen POS-tags

Om uiteindelijk de POS-tags toe te kunnen gebruiken als feature, is het nodig om ze eerst toe te voegen aan de dataset. Hierbij wordt gebruik gemaakt van twee POS-taggers, een Engelse en Nederlandse. Vervolgens worden 100 tweets handmatig geannoteerd om zo de accuratesse van de taggers te kunnen meten. Dit proces wordt beschreven in deze sectie.

De eerste stap is om de twee POS-taggers te ontwikkelen. Voor de Engelse tagger wordt de standaard POS-tagger gebruikt uit de NLTK-package[2]. De Nederlandse tagger is gemaakt met behulp van dit package, namelijk door de unigramtagger zie sectie 2.3. Er is gekozen voor een unigramtagger zodat er naar elk woord afzonderlijk kan worden gekeken. Hierdoor wordt het analyseren van data met code-switching eenvoudiger. De tagger wordt getraind op de Nederlandse dataset uit CoNLL2002, zie Tjong Kim Sang (2002). De dataset bestaat uit verschillende edities van de Belgische krant 'De Morgen' en is voorzien van POS-tags. Daarnaast wordt gebruik gemaakt van een defaulttagger, die dient als een back-off voor de unigramtagger. Deze defaulttagger rekent dezelfde tag toe aan elk woord en wordt alleen gebruikt wanneer de unigramtagger faalt (zie sectie 2.3). Omdat zelfstandige naamwoorden het meeste voorkomen in de trainingsdata, zal deze defaulttagger aan elk woord de tag van zelfstandig naamwoord toekennen.

Nu de twee verschillende taggers ontwikkeld zijn, kunnen ze worden ingezet. Hiertoe wordt de taalcategorie van elk token uit de tweets bekeken. Wanneer het in de categorie "Nederlands"valt, dan zal de Nederlandse tagger het woord van een POS-tag voorzien. De tagger kijkt dus niet naar de rest van de zin, slechts naar dit woord. Wanneer het woord in de categorie Engels of Gemixt valt zal de standaard Engelse tagger van NLTK dit woord van een POS-tag voorzien. De overige categorieën, Sociale Media Termen, Andere Taal en Onduidelijk, worden allen van de tag "Misc"(overig) voorzien.

Elke token is nu voorzien van een bepaalde POS-tag door de twee taggers. Om een indicatie voor de juistheid

[1] `http://illc.uva.nl/~raquel/data/CS_prediction.zip`
[2] `nltk.org`

van de tags te geven worden er uit de 1300 tweets 100 tweets handmatig van een POS-tags voorzien. Hiervoor worden 100 willekeurige tweets geselecteerd, die in totaal 1566 woorden bevatten. Deze tweets worden vervolgens met bijbehorende POS-tags getoond aan de annotator, die aangeeft of de POS-tags juist zijn. De annotator heeft als moedertaal Nederlands en spreekt vloeiend Engels en is daarom geschikt voor deze taak. Het percentage van het aantal goede POS-tags is 86.8% ± 0.9%, waaruit wordt geconcludeerd dat de POS-tags betrouwbaar zijn.

De POS-tags zijn dus toegevoegd aan de dataset door gebruik te maken van twee verschillende taggers: wanneer een woord Engels was werd de Engelse tagger ingezet, als het ging om een Nederlands woord de Nederlandse tagger. Vervolgens blijkt dat 86.8 ± 0.9% van de POS-tags goed is geclassificeerd, waaruit er geconcludeerd wordt dat deze tags betrouwbaar zijn.

.

3.3 Classificatie

De drie classificatiemodellen die worden gebruikt om code-switching te herkennen zijn: het Multinomial Naive Bayes model, het Support Vector Machine model en het Decision Tree model (voor uitleg zie 2). In deze sectie zal eerst de taak van de modellen worden omschreven, wat de input is en wat de gewenste output. Vervolgens zal worden ingegaan op de verschillende features die de modellen gebruiken. Een van die features zal de POS-tag van het woord zijn, om zo te onderzoeken of dit een positief effect heeft op de classificatie. De andere features zijn gekozen omdat ze worden gebruikt in de huidige classificatiemodellen, ontworpen door Dongen (2017).

3.3.1 De taak

De taak van de classificatiemodellen is het herkennen van code-switching in Nederlands-Engelse tweets. Hierbij gaat het om dat er wordt voorspelt of er een code-switch plaats vindt tussen de vorige en de huidige token. Er wordt alleen gefocust op intrasententiele code-switching, zie sectie 2.1. In deze scriptie worden alleen code-switches tussen Engels en Nederlands bestudeerd, de rest van de talen worden niet meegenomen. Hierbij worden de woorden die als taallabel Gemixt hebben, gerekend tot Engels, gelijk aan de aanpak van Dongen (2017). Daarnaast zijn er nog tags die neutraal zijn: Sociale Media Termen, Andere Taal en Onduidelijk: hierbij kan er dus geen code-switching plaats vinden.

Hieronder zijn twee voorbeelden van de taak weergeven. Hierbij staan ook de met de bijbehorende taallabels van de woorden. SMT staat voor Sociale Media Term. Een code-switch wordt met een * aangegeven. Er vindt dus geen code-switching plaats bij de wisseling van de Sociale Media Term naar Nederlands in het eerste voorbeeld, aangezien Sociale Media neutraal is.

@naam	beschouwend	en	in	the	line	of	fire
SMT	NL	NL	NL	*ENG	ENG	ENG	ENG
-1	-1	-1	-1	1	-1	-1	-1

Basic	*kan	dus	echt	heel	*stylish	*zijn
ENG	NL	NL	NL	NL	ENG	NL
-1	1	-1	-1	-1	1	1

Verder is er in het voorbeeld de gewenste output te zien, waarbij -1 staat voor "geen code-switch", en 1 staat voor "code-switch". Dit is de gewenste output van de classificatiemodellen, aangezien het doel is om code-switches te herkennen.

3.3.2 Features

Om de classificatiemodellen de punten van code-switching te laten herkennen is het nodig om verschillende features te selecteren. In deze sectie zullen de features die Dongen (2017) gebruikt in haar onderzoek worden besproken, aangezien deze ook in dit onderzoek zullen worden gebruikt. Ten slotte wordt de POS-tag feature toegelicht, die dit onderzoek toevoegt.

De features van Dongen (2017) maken alleen gebruik informatie die beschikbaar vóór de token waarvan wordt voorspeld of het een code-switch is. Wanneer er dus moet worden bepaald of het derde woord een code-switch is, wordt er informatie gebruikt van de eerste twee woorden, en niet van latere woorden. Hierdoor kan deze tool worden gebruikt in realtime applicaties (Dongen 2017). Aangezien in dit onderzoek bepaalde resultaten van Dongen (2017) worden gereproduceerd, zal dit ook het geval zijn in deze scriptie.

De features van Dongen (2017) maken gebruik van de variabele n. Hierbij staat n voor het laatste woord dat de huidige taal bepaalt. Aangezien er ook neutrale woorden voor kunnen komen, is n dus niet altijd de taal van het vorige woord.

Taal
De eerste feature gaat om de taal van n. Wanneer n Engels is (dus geclassificeerd is als Engels of Gemixt), krijgt het de waarde 2, als het Nederlands is de waarde 1, en wanneer n (nog) niet bestaat is de waarde 0. Deze informatie wordt verkregen door de taallabels die de woorden hebben gekregen. Hierbij moet worden opgemerkt dat dit dus een ideale situatie is, aangezien deze labels handmatig geannoteerd zijn.

De tweede feature is de taal van $n - 1$, dus het woord voorafgaand aan n. Hierbij gelden dezelfde waarden als bij de eerste feature. De onderbouwing van deze feature is dat de meeste Engelse woorden alleen voorkomen (53%) (Dongen 2017).

Lengte
Feature 3 is de lengte van n. Dongen (2017) gebruikte deze omdat dit mogelijk info over de POS-tag geeft, aangezien in het algemeen een lidwoord korter is dan een werkwoord. In dit onderzoek wordt een meer betrouwbare feature gebruikt voor de POS-tag, maar voor de volledigheid wordt deze feature ook gebruikt.

Aantal codeswitches
De overige features, 4, 5 en 6 tellen het aantal codeswitches tot en met n. Feature 4 telt de codes-switches van Nederlands naar Engels, feature 5 van Engels naar Nederlands en 6 telt het totaal aantal switches. Dit is een nuttige feature; wanneer er al een een code-switch is geweest naar het Engels, kan er niet nog een plaatsvinden (Dongen 2017).

POS-tags
De feature die dit onderzoek toevoegt is de POS-tag van het huidige woord. Hierbij wordt er dus informatie gebruikt van het huidige woord, en niet alleen van de woorden ervoor. De onderbouwing voor deze feature is gegeven in sectie 2.4.

Een overzicht van alle features is te vinden in tabel (1). Er zal gebruik gemaakt worden van drie verschillende feature sets. De baseline set bestaat alleen uit feature 1. Feature 1,4 en 6 vormen samen de tweede set, de zogeheten top-3 set. In Dongen (2017) kreeg deze combinatie van features de hoogste resultaten. Om deze reden zal deze ook in dit onderzoek van deze set gebruik worden gemaakt. De laatste set bestaat uit alle features, gelijk aan Dongen (2017). Aan de top-3 set en aan de gehele feature set zal telkens de POS-feature worden toegevoegd. Dan kan er worden vergeleken of de resultaten verbeteren door de POS-tag.

3.3.3 Trainen

De drie verschillende classificatiemodellen zijn getraind en getest door middel van 5-fold cross-validation. Er is hiervoor gekozen om optimaal gebruik te maken van de dataset. Hiervoor was eerst nodig om de data over de vijf subsets te verdelen. Zoals in sectie 3.1 is benoemd zijn de labels niet gelijk verdeeld. De Engelstalige woorden maken zijn slechts een klein percentage van de totale dataset (1.4%). Verder is er in 8.5% van de

Tabel 1: *In deze tabel is een kort overzicht van de features weergegeven.*

Feature nummer	Feature
1	Taal van n
2	Taal van $n-1$
3	Lengte van n
4	Aantal code-switches NL-EN tot en met n
5	Aantal code-switches EN-NL tot en met n
6	Totaal aantal code-switches tot en met n
7	POS-tag

tweets minstens één Engelstalig woord. Er is gekozen om ervoor te zorgen dat er in elke subset evenveel tweets met minstens één Engelstalig woord zijn. Zo worden er vergelijkbare testsets en trainingsets gegenereerd. Verder is de samenstelling van deze folds willekeurig. Dit is gedaan door gebruik te maken van een random seed. Voor zowel het Multinomial Naive Bayes model als het Decision Tree model wordt gebruik gemaakt van de package scikit-learn[3], versie 0.20.1. Voor het Support Vector Machine model wordt gebruik gemaakt van de package PyStruct[4].

3.3.4 Evaluatie

Wanneer de classificatiemodellen zijn getraind worden ze getest op een nieuwe dataset. Hierbij wordt het model geëvalueerd om zo de prestatie te meten. Als maat voor de prestatie kan de accuratesse niet worden gebruikt. Dit komt omdat de dataset scheef is: de Engelse woorden zijn slechts een klein percentage van de dataset. Hierdoor krijgt een model dat geen enkel switch-point voorspelt een accuratesse van 98%. Dit is onwenselijk, en daarom zullen de precision, recall en F1 van de modellen worden bepaald.

De precision is de proportie van het aantal goed voorspelde positieven gedeeld door alle voorspelde positieven, dus inclusief de foutieve positieven. Recall is het percentage van de goed voorspelde positieven gedeeld door het totaal aantal positieven. In deze context is de recall dus het aantal goed voorspelde code-switches gedeeld door het aantal echte code-switches.

De F1-score is het harmonisch gemiddelde van de precision en recall. De F1-score wordt vaak gebruikt om de prestaties van classificatiemodellen te meten. Voor alle drie de waarden geldt dat 1 de best haalbare prestatie is, en 0 de slechtste.

[3] `https://scikit-learn.org/stable/index.html`
[4] `https://pystruct.github.io/`

4 Resultaten

In deze sectie zullen de resultaten per classificatiemodel worden gepresenteerd. Het overzicht van de volledige resultaten staat in de appendix (A), hierbij worden ook de scores van de modellen met de baseline feature set weergegeven. In de discussie (5) worden de resultaten vergeleken met de resultaten van de huidige modellen van Dongen (2017).

4.1 Multinomial Naive Bayes model

In tabel 2 staan de belangrijkste resultaten weergegeven van het Multinomial Naive Bayes model (MNB). Hierin werd gebruik gemaakt van de drie beste features van Dongen, namelijk feature 1, 4 en 6 (zie 3.3.2). Wanneer de MNB gebruik maakt van deze drie features en de POS-feature, dan wordt de precision iets hoger. Dit verschil is volgens een gepaarde t-test met $p < 0.05$ niet significant. De recall en ook de F1-score zijn ook hoger wanneer deze feature is toegevoegd, en dit verschil is wel significant volgens een gepaarde t-test met $p < 0.01$. Dus voor het MNB model verbetert het model significant wanneer de POS-tags worden toegevoegd aan de top 3 features.

Tabel 2: *In deze tabel zijn de precision, recall en F1-score weergegeven voor het Multinomial Naive Bayes model. Ook de standaarddeviaties zijn gerapporteerd. In de eerste rij maakt dit model maakt gebruik van de Top-3 features, in de tweede rij staan de resultaten voor de Top-3 features inclusief de POS-feature. De resultaten met de beste F1-score zijn dikgedrukt.*

Features	Precision	Recall	F1-score
Top 3 (1,4,6)	0.117 ± 0.105	0.060 ± 0.067	0.075 ± 0.080
Top 3 en POS (1,4,6,7)	$\mathbf{0.138 \pm 0.054}$	$\mathbf{0.295 \pm 0.084}$	$\mathbf{0.187 \pm 0.065}$

Bij het MNB-model is ook geëxperimenteerd met de gehele feature set, zie tabel 3. Hierbij is het model eerst getest met feature 1 tot en met 6 en vervolgens met feature 1 tot en met 7. De precision, recall en de F1-score verbeteren significant wanneer de POS-feature wordt toegevoegd, volgens een gepaarde t-toets met $p < 0.01$.

Tabel 3: *In deze tabel zijn de precision, recall en F1-score weergegeven voor het Multinomial Naive Bayes model wanneer er gebruik gemaakt wordt van de gehele feature set. Ook de standaarddeviaties zijn gerapporteerd. In de eerste rij maakt dit model maakt gebruik van de features 1 tot en met 6, in de tweede rij staan de resultaten voor deze features inclusief de POS-feature. De resultaten met de beste F1-score zijn dikgedrukt.*

Features	Precision	Recall	F1-score
Features 1-6	0.061 ± 0.050	0.126 ± 0.078	0.082 ± 0.061
Features 1-7	$\mathbf{0.129 \pm 0.048}$	$\mathbf{0.293 \pm 0.086}$	$\mathbf{0.177 \pm 0.059}$

4.2 Decision Tree classificatiemodel

In tabel 4 zijn een deel van de resultaten van het Decision Tree model weergegeven. Hierbij is gebruik gemaakt van top-3 featureset. Wanneer aan de top-3 features ook de POS-feature wordt toegevoegd, verbeteren zowel de precision, recall en F1-score. Voor alle drie is dit verschil significant volgens een gepaarde t-toets met $p < 0.01$.

Tabel 4: *In deze tabel zijn de precision, recall en F1-score weergegeven voor het Decision Tree model. Ook de standaarddeviaties zijn gerapporteerd. In de eerste rij maakt dit model maakt gebruik van de Top-3 features, in de tweede rij staan de resultaten voor de Top-3 features inclusief de POS-feature. De resultaten met de beste F1-score zijn dikgedrukt.*

Features	Precision	Recall	F1-score
Top 3 (1,4,6)	0.100 ± 0.137	0.009 ± 0.012	0.016 ± 0.021
Top 3 en POS (1,4,6,7)	$\mathbf{0.547 \pm 0.101}$	$\mathbf{0.263 \pm 0.042}$	$\mathbf{0.350 \pm 0.044}$

In tabel 5 zijn de resultaten van het Decision Tree model weergegeven wanneer alle features worden gebruikt, dus 1 tot en met 6. Wanneer bovenop deze features ook de POS-feature wordt toegevoegd verbeteren alle drie de scores. Echter, deze verschillen zijn niet significant volgens een gepaarde t-test met $p < 0.05$.

Tabel 5: *In deze tabel zijn de precision, recall en F1-score weergegeven voor het Decision Tree model wanneer er gebruik gemaakt wordt van de gehele feature set. Ook de standaarddeviaties zijn gerapporteerd. In de eerste rij maakt dit model maakt gebruik van de features 1 tot en met 6, in de tweede rij staan de resultaten voor deze features inclusief de POS-feature. De resultaten met de beste F1-score zijn dikgedrukt.*

Features	Precision	Recall	F1-score
Features 1-6	0.481 ± 0.173	0.162 ± 0.039	0.240 ± 0.066
Features 1-7	$\mathbf{0.552 \pm 0.143}$	$\mathbf{0.204 \pm 0.068}$	$\mathbf{0.294 \pm 0.084}$

4.3 Support Vector Machine

In tabel 6 zijn de resultaten van het Support Vector Machine model weergeven, waar gebruik wordt gemaakt van de top-3 feature set. Het model heeft nul echte positieven, waardoor de precision, recall en F1 score 0.000 zijn. Wanneer de POS-feature wordt toegevoegd aan de set verbeteren de drie scores. Deze verschillen zijn significant volgens een gepaarde t-test met $p < 0.01$.

Tabel 6: *In deze tabel zijn de precision, recall en F1-score weergegeven voor het Support Vector Machine model. Ook de standaarddeviaties zijn gerapporteerd. In de eerste rij maakt dit model maakt gebruik van de Top-3 features, in de tweede rij staan de resultaten voor de Top-3 features inclusief de POS-feature. De resultaten met de beste F1-score zijn dikgedrukt.*

Features	Precision	Recall	F1-score
Top 3 (1,4,6)	0.000	0.000	0.000
Top 3 en POS(1,4,6,7)	$\mathbf{0.292 \pm 0.095}$	$\mathbf{0.093 \pm 0.030}$	$\mathbf{0.139 \pm 0.039}$

Ten slotte is het SVM model ook getest met alle features. Ook dit model heeft nul echte positieven, waardoor de scores nul zijn. De scores verbeteren significant wanneer de POS-feature wordt toegevoegd. De significantie is bepaald door middel van een gepaarde t-test met $p < 0.01$.

Tabel 7: *In deze tabel zijn de precision, recall en F1-score weergegeven voor het Support Vector Machine model wanneer er gebruik gemaakt wordt van de gehele feature set. Ook de standaarddeviaties zijn gerapporteerd. In de eerste rij maakt dit model maakt gebruik van de features 1 tot en met 6, in de tweede rij staan de resultaten voor deze features inclusief de POS-feature. De resultaten met de beste F1-score zijn dikgedrukt.*

Features	Precision	Recall	F1-score
Features 1-6	0.000	0.000	0.000
Features 1-7	$\mathbf{0.351 \pm 0.044}$	$\mathbf{0.144 \pm 0.052}$	$\mathbf{0.199 \pm 0.061}$

5 Discussie

In deze sectie zullen allereerst de resultaten van dit onderzoek kort worden besproken. Vervolgens worden deze resultaten vergeleken met de resultaten van Dongen (2017), om zo de verschillen te benoemen en te verklaren. Daarna worden beperkingen van dit onderzoek genoemd, om zo te bespreken wat mogelijke verbeteringen zijn, en worden er suggesties gedaan voor vervolgonderzoek.

5.1 Bespreking resultaten

Uit de tabellen 2 tot en met 7 blijkt dat in alle gevallen de F1-score, die een combinatie is van de precision en recall, hoger wordt wanneer POS-tags in de feature zitten. In de meeste gevallen wordt zowel de precision als de recall hoger. In een enkel geval (MNB model in tabel 2) blijft de precision nagenoeg gelijk, maar verbetert de recall significant.

Omdat de F1-score de precision en recall combineert, zal voor het bepalen van het beste model naar deze waarden worden gekeken. De hoogste score wordt bereikt bij het Decision Tree model waarbij gebruik wordt gemaakt van de top-3 feature set inclusief POS-feature. De F1-score van dit model is 0.350. Dit verschilt echter niet significant van de F1-score van het Decision Tree model met de gehele feature set, deze is namelijk 0.294.

De grootste verbetering van de F1-score door het toevoegen van de POS-feature wordt ook behaald door het Decision Tree Model met features 1,4,6 en 7.

5.2 Vergelijking met Dongen (2017)

In dit onderdeel zullen de verschillen met Dongen (2017) worden uitgelicht en verklaard. In tabel 8 staan de belangrijkste resultaten van Dongen (2017) samengevat. De resultaten zullen per model worden vergeleken, beginnend met het Multinomial Naive Bayes model. De resultaten van het MNB model met de top 3 feature set verschillen weinig met de resultaten uit dit onderzoek (tabel 2). Zowel de precision als de recall zijn nagegoeg gelijk, waardoor de F1-score ook weinig verschilt. Echter, er is wel een verschil in de resultaten van het MNB model met de gehele feature set. Uit tabel 3 blijkt dat de precision uit dit onderzoek (0.061) veel lager is dan het resultaat (0.188) van Dongen (2017). De recall is wel ongeveer gelijk.

Bij het Decision Tree model verschillen de resultaten van Dongen (2017) hevig met dit onderzoek. Wanneer de resultaten uit tabel 4 met tabel 8 worden vergeleken, valt vooral op dat de scores uit dit onderzoek veel lager zijn. Dit geldt voor zowel de precision als voor recall, en dus ook voor de F1-score. Hierbij moet worden opgemerkt dat de standaardafwijkingen voor de scores ook redelijk hoog zijn. Ook Dongen (2017) rapporteerde redelijk grote standaardafwijkingen. Hierdoor zou het kunnen dat de scores dichter bij elkaar liggen. Wanneer alle features worden gebruikt zijn de verschillen bij het Decision Tree model een stuk kleiner. De precision uit tabel 8 is 0.425 en ligt dichtbij de waarde uit tabel 5 (0.481). De recall van dit onderzoek is wel veel lager (0.162) dan de recall van Dongen (2017) (0.415).

Ten slotte worden de resultaten van het Support Vector Machine model met elkaar vergeleken. Voor de Top 3 feature set zijn de scores gelijk, ook bij Dongen (2017) waren alle scores 0.000. Voor de gehele feature set is er wel verschil tussen de scores. In dit onderzoek waren alle waarden 0.000 (tabel 7). Bij Dongen (2017) zijn deze waarden iets hoger, met name bij de precision is er een groot verschil. Deze is bij Dongen (2017) namelijk 0.134.

Uiteindelijk moet worden opgemerkt dat de beste resultaten van Dongen (2017) hoger zijn dan de best behaalde resultaten uit dit onderzoek. Echter, er wordt wel een hogere precision dan de hoogste precision van Dongen (2017) behaald, bij beide Decision Tree modellen. Zowel als bij Dongen (2017) en dit onderzoek behaalt het Decision Tree model de beste resultaten.

Tabel 8: *In deze tabel zijn de precision (P), recall (R) en F1-score (F) weergegeven zoals gerapporteerd in Dongen (2017). Alle drie de modellen staan in deze tabel: Multinomial Naive Bayes (MNB), Decision Tree en Support Vector Machines (SVM). De resultaten van de top-3 feature set zijn weergegeven, alsmede van features 1-6 (Alles). Aangezien niet alle standaardafwijkingen gerappoteerd zijn in Dongen (2017) zijn deze ook niet in de tabel weergegeven.*

	MNB			Decision Tree			SVM		
Features	P	R	F1	P	R	F1	P	R	F1
Top 3 (1,4,6)	0.152	0.074	0.096	**0.439**	**0.507**	**0.463**	0.000	0.000	0.000
Alles (1-6)	0.188	0.102	0.124	0.425	0.415	0.416	0.134	0.008	0.016

5.3 Verklaringen verschillen

In sommige gevallen verschillen de resultaten uit Dongen (2017) hevig met dit onderzoek. In de meeste gevallen zijn in dit onderzoek de scores lager dan in het onderzoek van Dongen (2017). Er zullen nu enkele verklaringen voor deze verschillen worden gegeven.

Ten eerste kan de methode van de twee onderzoeken verschillen. Er is geprobeerd om nauwgezet de methode van Dongen (2017) te volgen, maar niet elke stap werd in haar onderzoek beschreven. Hierdoor moesten er in dit onderzoek ook keuzes worden gemaakt die mogelijk de resultaten hebben beïnvloed. In deze paragraaf worden deze verschillen kort benoemd.

Het eerste mogelijke verschil gaat om het gelijk verdelen van de taallabels over de verschillende folds, zie 3.3.3. Aangezien de taallabels scheef verdeeld zijn over de dataset, was het nodig om ervoor te zorgen dat de folds dezelfde verdeling hadden. Het is namelijk nodig dat de trainingsdata representatief is voor de testdata. Er is in deze scriptie gekozen om in elke fold evenveel tweets te hebben die minstens één Engelse token bevatten. Deze keuze had ook anders kunnen zijn, door bijvoorbeeld ervoor te kiezen dat in elke fold evenveel Engelse woorden zijn. Een andere keuze was geweest om ervoor te zorgen dat elke fold evenveel code-switches bevat. Op dit punt was het onderzoek van Dongen (2017) ambigu, dus was het nodig om hier zelf een keuze in te maken. Hierdoor kunnen de resultaten verschillend zijn. Daarnaast heeft Dongen (2017) ervoor gekozen om expliciet elke taallabel evenredig te verspreiden over de verschillende folds. In deze scriptie is dit alleen voor het Engelse taallabel gedaan.

Een ander verschil in de methode kan zijn dat Dongen (2017) andere waarden aan bepaalde features heeft gegeven. In sectie 3.3.2 is uiteengezet dat bij feature 1 en 2 de waarde 0, 1 of 2 is. In het onderzoek van (Dongen 2017) wordt vermeldt dat deze waardes -1, 0 of 1 zijn. In het MNB model, dat Dongen (2017) ook gebruikt, is het echter onmogelijk om negatieve getallen als featurewaardes te gebruiken. De reden hiervoor is dat een multinomiale distributie geen negatieve waarden kan bevatten. Echter, dit verschil in methode maakt waarschijnlijk weinig uit voor de verschillen in resultaten.

Dongen (2017) heeft niet altijd alle specificaties van de modellen genoemd, waardoor het kan dat er toch andere resultaten zijn. Er is wel gebruik gemaakt van precies dezelfde packages voor de modellen (sectie 3.3.3), maar de precieze invulling hiervan kan verschillen.

Het laatste verschil is dat (Dongen 2017) slechts 90% van haar dataset gebruikt, om zo nog een testset te hebben voor een latere opdracht. Omdat dit onderzoek niet nog een testset nodig had, is ervoor gekozen om alle tweets te gebruiken.

Een andere verklaring voor de verschillende resultaten van Dongen (2017) en dit onderzoek is dat er een component van willekeurigheid in zit. Voor dit onderzoek is er een random seed gebruikt om de tweets over de verschillende folds te verdelen, zoals beschreven bij sectie 3.3.3. Wanneer er een andere random seed werd gebruikt, kregen de folds een andere verdeling. Hierbij werd nog wel gezorgd dat elke fold dezelfde hoeveelheid (deels) Engelse tweets bevatte. Wanneer er een andere seed werd gebruikt, waren de resultaten verschillend. Dit viel vooral op bij classificatiemodellen met lage scores; wanneer het aantal echte positieven erg laag was, konden de vijf testuitslagen nogal erg van elkaar verschillen. Dit is ook te zien aan de grote standaardafwijkingen wanneer een model een lage precision en recall heeft, zoals het geval is bij de Decision

Tree in tabel 4. Deze willekeur kan de verschillen ook deels verklaren.

5.4 Beperkingen

In deze sectie zullen de beperkingen en de verbeteringen van dit onderzoek worden besproken. De eerste tekortkoming van dit onderzoek is dat het is uitgevoerd in een ideale setting. De taallabels van de data zijn met de hand geannoteerd: hierdoor hadden de modellen alle correcte informatie. Dit beïnvloedt features, die nu (bijna) altijd correct zijn. In de praktijk zullen bijvoorbeeld de eerste twee features(zie sectie 3.3.2)) niet altijd correct zijn. Deze features baseren zich op de taal van n en $n-1$, die hoogstwaarschijnlijk beter worden bepaald door een menselijke annotator dan door een model. Wanneer de modellen dus in praktijk worden ingezet, zullen de resultaten hoogstwaarschijnlijk minder goed zijn.

Om de prestaties van de POS-tagger te meten zijn 100 tweets door één persoon voorzien van POS-tags. Hierbij zou het goed kunnen dat deze persoon verschillende fouten heeft gemaakt. Een verbetering zou zijn om een andere persoon de tags laten te controleren, om zo te kijken of ze in overeenstemming zijn.

Een andere tekortkoming is dat er alleen geëxperimenteerd is met de POS-tag van het woord waarin de code-switch zich bevindt. Er zijn echter ook aanwijzingen dat de POS-tag van woord voorafgaand aan de switch ook nuttig is als feature (Soto, Cestero en Hirschberg 2018). Ook is er niet geëxperimenteerd met andere combinaties van features of andere modellen. De reden hiervoor is dat dit niet paste binnen de omvang van het project.

Een laatste tekortkoming is dat in de dataset weinig code-switches waren, waardoor er minder informatie voor de modellen beschikbaar was. In een fold waren er gemiddeld 41 code-switches en 3893 woorden, wat betekent dat er in ongeveer 1% van de woorden een code-switch plaatsvindt. Hierdoor hadden de modellen weinig informatie om code-switches te voorspellen. In het vervolg is het aangeraden om een dataset te gebruiken waarbij meer code-switches plaatsvinden, om de modellen daarop te trainen. Een andere optie is om een grotere dataset te gebruiken.

5.5 Vervolgonderzoek

Er zijn verschillende voor vervolgonderzoeken mogelijk. Allereerst zou er geëxperimenteerd kunnen worden met andere modellen, zoals een neuraal netwerk. Dit wordt al gedaan in het onderzoek van Adel, Vu, Kraus e.a. (2013) en Attia e.a. (2019). Beide onderzoeken laten zien dat ook bij neurale netwerken POS-tags van toegevoegde waarde zijn. Bovendien hebben deze modellen veelbelovende resultaten. In het onderzoek van (J. C. Chang en C. Lin 2014) hebben de neurale netwerken hogere resultaten dan het SVM-model. Voor een neuraal netwerk is het wel noodzakelijk dat er meer data beschikbaar is. Een ander mogelijk model is het Factored Language model, dat wordt gebruikt in combinatie met POS-tags in Adel, Vu, Kirchhoff e.a. (2015). Aangezien beide modellen tot goede resultaten leiden bij andere talen, zou het interessant zijn om te onderzoeken of het ook goed werkt voor het Nederlands-Engels.

Daarnaast kan ook worden geëxperimenteerd met verschillende combinaties van features. Ook is het mogelijk om nieuwe features toe te voegen. Zoals in de vorige sectie is besproken, is het ook nuttig om naar de POS-tag van het woord voorafgaand aan de code-switch te kijken (Soto, Cestero en Hirschberg 2018). Deze, en andere features, zouden ook een positieve bijdrage kunnen leveren bij het herkennen van Nederlands-Engelse code-switching.

Vervolgens is het nodig om te onderzoeken hoe er POS-tags toegevoegd kunnen worden aan een dataset die code-switching bevat. In deze geïdealiseerde setting was het mogelijk om de taallabels te gebruiken, maar juist bij een tekst met meerdere talen is het moeilijk om POS-labels toe te voegen (Çetinoğlu, Schulz en Vu 2016). Hiermee is al geëxperimenteerd in andere talen zoals Engels-Iers en Engels-Hindi (Lynn en Scannell 2019; Vyas e.a. 2014), maar staat voor het Engels-Nederland nog weinig gedaan.

Ten slotte is het interessant om te kijken naar andere talen zoals Nederlands-Turks of Nederlands-Fries. Ook kunnen misschien andere datasets worden gebruikt, die bijvoorbeeld uit gesproken taal afkomstig zijn of uit een ander sociaal medium. Al deze onderzoeken kunnen uiteindelijk bijdragen om het herkennen van code-switching te verbeteren.

6 Conclusie

Het doel van deze scriptie was om antwoord te geven op de volgende vraag: "*Hoe kunnen de classificatiemodellen voor het herkennen van code-switching in Nederlands-Engelse tweets worden verbeterd?*". De deelvraag luidde: "*Wat is de invloed van het toevoegen van POS-tags op de F1-score van de classificatiemodellen?*". Hiertoe is gebruik gemaakt van een dataset bestaande uit Nederlands-Engelse tweets, gemaakt door Dongen (2017). Aan deze dataset zijn POS-tags toegevoegd door gebruik te maken van twee verschillende taggers, een Nederlandse en een Engelse. Vervolgens is de aanpak van Dongen (2017) deels gevolgd: dezelfde features zijn gekozen en ook dezelfde classificatiemodellen zijn onderzocht. Daarnaast is de nieuwe POS-feature toegevoegd aan elk van de modellen. De classificatiemodellen die zijn gebruikt zijn het Multinomial Naive Bayes model, het Decision Tree model en het Support Vector Machine model. Hierbij bleek dat in alle gevallen de F1-score verbeterde. De grootste verbetering was bij het Decision Tree model, wanneer gebruik gemaakt werd van de top 3 feature set inclusief de POS-tag. De beste F1-score werd ook behaald door dit model.

Wanneer dit onderzoek wordt vergeleken met het onderzoek van Dongen (2017) zijn er bij voornamelijk het Decision Tree model en het Support Vector Machine model grote verschillen. Deze kunnen worden verklaard door een verschil in methode en door de willekeur van de trainingsdata.

In alle gevallen leidde de toevoeging van de POS-feature tot een verhoging van de F1-score, waaruit blijkt dat het toevoegen van deze feature een positieve invloed heeft op de prestaties van de modellen. De classificatiemodellen voor het herkennen van code-switching kunnen dus verbeterd worden door het gebruik van POS-tags. Deze bevinding is in lijn met de eerdere onderzoeken naar de relatie tussen het herkennen van code-switching en POS-tags.

Referenties

Abu Mostafa, Yaser S., Malik Magdon-Ismail en Hsuan-Tien Lin (2012). *Learning from data: A short course*. AMLbook.com.

Adel, Heike, Ngoc Thang Vu, Katrin Kirchhoff e.a. (2015). „Syntactic and semantic features for code-switching factored language models". In: *IEEE/ACM transactions on audio, speech, and language Processing* 23.3, p. 431–440.

Adel, Heike, Ngoc Thang Vu, Franziska Kraus e.a. (2013). „Recurrent neural network language modeling for code switching conversational speech". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, p. 8411–8415.

Attia, Mohammed e.a. (2019). „POS Tagging for Improving Code-Switching Identification in Arabic". In: *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, p. 18–29.

Barman, Utsab e.a. (2014). „Dcu-uvt: Word-level language classification with code-mixed data". In: *Proceedings of the First Workshop on Computational Approaches to Code Switching*, p. 127–132.

Bhat, Irshad Ahmad e.a. (2018). *Universal Dependency Parsing for Hindi-English Code-switching*. arXiv: `1804.05868 [cs.CL]`.

Bokamba, Eyamba G (1989). „Are there syntactic constraints on code-mixing?" In: *World Englishes* 8.3, p. 277–292.

Brants, Thorsten (2000). „TnT-a statistical part-of-speech tagger". In: *arXiv preprint cs/0003055*.

Çetinoğlu, Özlem (2016). „A Turkish-German code-switching corpus". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, p. 4215–4220.

Çetinoğlu, Özlem, Sarah Schulz en Ngoc Thang Vu (2016). *Challenges of Computational Processing of Code-Switching*. arXiv: `1610.02213 [cs.CL]`.

Chang, Joseph Chee en Chu-Cheng Lin (2014). „Recurrent-Neural-Network for Language Detection on Twitter Code-Switching Corpus". In: *CoRR* abs/1412.4314. arXiv: `1412.4314`. URL: `http://arxiv.org/abs/1412.4314`.

Claeser, Daniel, Dennis Felske en Samantha Kent (2017). „Token level code-switching detection using Wikipedia as a lexical resource". In: *International Conference of the German Society for Computational Linguistics and Language Technology*. Springer, p. 192–198.

Claros, Monica Stella Cardenas en Neny Isharianty (2009). „Code-Switching and Code-Mixing in Internet Chatting: between 'yes,''ya', and 'si': a Case Study". In: *Jaltcall* 5, p. 67–78.

Cutting, Douglass e.a. (1992). „A practical part-of-speech tagger". In: *Third Conference on Applied Natural Language Processing*, p. 133–140.

Das, Amitava en Björn Gambäck (2015). „Code-mixing in social media text: the last language identification frontier?" In: *Revue TAL* 54(3), p. 41–46.

Dongen, N. (2017). „Analysis and Prediction of Dutch-English Code-switching in Dutch Social Media Messages". Masterscriptie. Universiteit van Amsterdam.

Eskander, Ramy e.a. (2014). „Foreign words and the automatic processing of Arabic social media text written in Roman script". In: *Proceedings of The First Workshop on Computational Approaches to Code Switching*, p. 1–12.

El-Haj, Mahmoud, Paul Rayson en Mariam Aboelezz (2018). „Arabic dialect identification in the context of bivalency and code-switching". In: *Proceedings of the 11th International Conference on Language Resources and Evaluation, Miyazaki, Japan*. European Language Resources Association, p. 3622–3627.

Hopper, Paul J en Sandra A Thompson (1984). „The discourse basis for lexical categories in universal grammar". In: *Language* 60.4, p. 703–752.

Jaech, Aaron e.a. (2016). „A neural model for language identification in code-switched tweets". In: *Proceedings of The Second Workshop on Computational Approaches to Code Switching*, p. 60–64.

Jain, Naman en Riyaz Ahmad Bhat (2014). „Language identification in code-switching scenario". In: *Proceedings of the First Workshop on Computational Approaches to Code Switching*, p. 87–93.

Joshi, Aravind (1982). „Processing of sentences with intra-sentential code-switching". In: *Coling 1982: Proceedings of the Ninth International Conference on Computational Linguistics*.

Kent, Samantha en Daniel Claeser (2019). „Incorporating Code-Switching and Borrowing in Dutch-English Automatic Language Detection on Twitter". In: *Proceedings of the Future Technologies Conference (FTC) 2018*. Red. door Kohei Arai, Rahul Bhatia en Supriya Kapoor. Cham: Springer International Publishing, p. 418–434. ISBN: 978-3-030-02686-8.

Kim, Seungbeom en Jongsoo Park (2007). *Automatic detection of character encoding and language*. Tech. rap. Tech. rep., Stanford University.

Lynn, Teresa en Kevin Scannell (2019). „Code-switching in Irish tweets: A preliminary analysis". In: *Proceedings of the Celtic Language Technology Workshop*, p. 32–40.

Mabule, D R (2015). „What is this? Is It Code Switching, Code Mixing or Language Alternating?" In: *Journal of Educational and Social Research* 5.1. ISSN: 2240-0524. URL: `https://www.mcser.org/journal/index.php/jesr/article/view/5628`.

Mahata, Sainik Kumar e.a. (2020). „Analyzing code-switching rules for english–hindi code-mixed text". In: *Emerging Technology in Modelling and Graphics*. Springer, p. 137–145.

Nguyen, Dong en Leonie Cornips (2016). „Automatic detection of intra-word code-switching". In: *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, p. 82–86.

Owoputi, Olutobi e.a. (2013). „Improved part-of-speech tagging for online conversational text with word clusters". In: *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies*, p. 380–390.

Poplack, Shana (2000). „Sometimes I'll start a sentence in Spanish y termino en español: Toward a typology of code-switching". In: *The bilingualism reader* 18.2, p. 221–256.

Rijhwani, Shruti e.a. (2017). „Estimating code-switching on twitter with a novel generalized word-level language detection technique". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, p. 1971–1982.

Samih, Younes e.a. (2016). „Multilingual code-switching identification via lstm recurrent neural networks". In: *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, p. 50–59.

Solorio, Thamar, Elizabeth Blair e.a. (2014). „Overview for the first shared task on language identification in code-switched data". In: *Proceedings of the First Workshop on Computational Approaches to Code Switching*, p. 62–72.

Solorio, Thamar en Yang Liu (2008). „Learning to predict code-switching points". In: *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, p. 973–981.

Soto, Victor, Nishmar Cestero en Julia Hirschberg (2018). „The Role of Cognate Words, POS Tags and Entrainment in Code-Switching." In: *Interspeech*, p. 1938–1942.

Tjong Kim Sang, Erik F. (2002). „Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition". In: *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*. URL: `https://www.aclweb.org/anthology/W02-2024`.

Vyas, Yogarshi e.a. (2014). „Pos tagging of english-hindi code-mixed social media content". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 974–979.

Xia, Meng Xuan (2016). „Codeswitching language identification using subword information enriched word vectors". In: *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, p. 132–136.

Zampieri, Marcos e.a. (2014). „A report on the DSL shared task 2014". In: *Proceedings of the first workshop on applying NLP tools to similar languages, varieties and dialects*, p. 58–67.

A Appendix: Alle Resultaten

Tabel 9: Multinomial Naive Bayes model

Features	Precision	Recall	F1-score
Baseline (1)	0.000	0.000	0.000
Top 3 (1,4,6)	0.117 ± 0.105	0.060 ± 0.067	0.075 ± 0.080
Top 3 en POS (1,4,6,7)	$\mathbf{0.138 \pm 0.054}$	$\mathbf{0.295 \pm 0.084}$	$\mathbf{0.187 \pm 0.065}$
Features 1-6	0.061 ± 0.050	0.126 ± 0.078	0.082 ± 0.061
Features 1-7	0.129 ± 0.048	0.293 ± 0.086	0.177 ± 0.059

Tabel 10: Decision Tree model

Features	Precision	Recall	F1-score
Baseline (1)	0.000	0.000	0.000
Top 3 (1,4,6)	0.100 ± 0.137	0.009 ± 0.012	0.016 ± 0.021
Top 3 en POS (1,4,6,7)	$\mathbf{0.547 \pm 0.101}$	$\mathbf{0.263 \pm 0.042}$	$\mathbf{0.350 \pm 0.044}$
Features 1-6	0.481 ± 0.173	0.162 ± 0.039	0.240 ± 0.066
Features 1-7	0.552 ± 0.143	0.204 ± 0.068	0.294 ± 0.084

Tabel 11: Support Vector Machine model

Features	Precision	Recall	F1-score
Baseline (1)	0.000	0.000	0.000
Top 3 (1,4,6)	0.000	0.000	0.000
Top 3 en POS(1,4,6,7)	0.292 ± 0.095	0.093 ± 0.030	0.139 ± 0.039
Features 1-6	0.000	0.000	0.000
Features 1-7	$\mathbf{0.351 \pm 0.044}$	$\mathbf{0.144 \pm 0.052}$	$\mathbf{0.199 \pm 0.061}$