

Scholieren met dyslexie van de derde graad middelbaar onderwijs ondersteunen bij het intensief lezen van wetenschappelijke artikelen via geautomatiseerde en gepersonaliseerde tekstvereenvoudiging.

De opbouw van een LLM-gedreven prototype voor geautomatiseerde en gepersonaliseerde tekstvereenvoudiging.

Dylan Cluyse.

Scriptie voorgedragen tot het bekomen van de graad van
Professionele bachelor in de toegepaste informatica

Promotor: Mevr. L. De Mol

Co-promotor: J. Decorte; J. Van Damme;

Academiejaar: 2022–2023

Eerste examenperiode

Departement IT en Digitale Innovatie .

**HO
GENT**

Woord vooraf

Deze scriptie en het bijhorende onderzoek zou niet tot stand zijn gekomen zonder de waardevolle bijdragen van diverse individuen die mij hebben ondersteund en gestimuleerd tijdens mijn onderzoek. Ik wil graag mijn oprechte dank betuigen aan deze personen.

Ten eerste, wil ik mijn promotor Lena De Mol bedanken voor haar uitmuntende begeleiding tijdens het onderzoek. Haar affiniteit voor technologie, taal en onderwijs heeft een perfecte match gevormd met mijn onderzoeksgebied. Daarnaast wil ik graag Johan Decorte en Jana Van Damme bedanken voor hun deskundige inbreng op de vakgebieden. Elke wekelijkse sessie met Johan bracht nieuwe inzichten in hoe ik het technologische component van mijn onderzoek kon aanpakken. Dit heeft mijn ambitie alleen maar vergroot. Ook wil ik Jana Van Damme bedanken voor haar begeleiding en *follow-up* op het gebied van logopedie. Haar expertise heeft mijn horizon verbreed in dit vakgebied. Ik wil ook graag Emmanuel Vercruysse en Johannes Nijs van Hogeschool Vives en Sofie Smet en Sophie Vyncke van Arteveldehogeschool bedanken voor hun bijdragen aan de referentieteksten voor het experiment. Emmanuel en Sophie hebben mij met veel plezier geholpen, ondanks hun drukke agenda's, en Johannes en Sofie hebben de taak op zich genomen tijdens de paasvakantie. Tot slot, wil ik mijn goede vriendin Lobke bedanken voor haar constante steun en aanmoediging tijdens het hele onderzoeksproces, alsook mijn grootste steunpunt die ik tijdens de opleiding heb leren kennen.

Ik wil graag benadrukken dat deze personen van onschatbare waarde zijn geweest voor het succes van mijn onderzoek en mijn eindresultaat. Hun inzet en toewijding hebben ertoe bijgedragen dat ik deze scriptie met trots kan presenteren.

Samenvatting

Ingewikkelde woordenschat en zinsbouw hinderen scholieren met dyslexie in het derde graad middelbaar onderwijs bij het lezen van wetenschappelijke artikelen. Gepersonaliseerde tekstvereenvoudiging helpt deze scholieren bij hun leesbegrip. Daarnaast kan artificiële intelligentie (AI) dit proces automatiseren om de werkdruk bij leraren en scholieren te verminderen. Dit onderzoek achterhaalt met welke technologische en logopedische aspecten AI-ontwikkelaars rekening moeten houden bij de ontwikkeling van een AI-toepassing voor geautomatiseerde en gepersonaliseerde tekstvereenvoudiging. Hiervoor is de volgende onderzoeksvraag opgesteld: "Hoe kan een wetenschappelijk artikel automatisch worden vereenvoudigd, gericht op de unieke noden van scholieren met dyslexie in het derde graad middelbaar onderwijs?". Een requirementsanalyse achterhaalt de benodigde functionaliteiten om gepersonaliseerde en geautomatiseerde tekstvereenvoudiging mogelijk te maken. Vervolgens wijst de vergelijkende studie uit welk taalmodel kan worden ingezet om de taak van gepersonaliseerde en geautomatiseerde tekstvereenvoudiging mogelijk te maken. De requirementsanalyse wijst uit dat toepassingen om wetenschappelijke artikelen te vereenvoudigen, gemaakt zijn voor een centrale doelgroep en geen rekening houden met de unieke noden van een scholier met dyslexie in het derde graad middelbaar onderwijs. Adaptieve software voor geautomatiseerde tekstvereenvoudiging is mogelijk, maar ontwikkelaars moeten meer inzetten op de unieke noden van deze scholieren.

Inhoudsopgave

Lijst van figuren	vii
--------------------------	------------

Lijst van tabellen	viii
---------------------------	-------------

1 Inleiding	1
1.1 Probleemstelling	2
1.2 Onderzoeksvraag	3
1.3 Onderzoeksdoelstelling	5
1.4 Opzet van deze bachelorproef	5
2 Stand van zaken	6
2.1 Inleiding	6
2.2 Specifieke noden en richtpunten	6
2.2.1 Specifieke noden van scholieren met dyslexie in de derde graad van het middelbaar onderwijs	7
2.2.2 Specifieke kenmerken van wetenschappelijke artikelen	8
2.3 Aanpakken voor tekstvereenvoudiging	12
2.3.1 Manuele tekstvereenvoudiging	12
2.3.2 Bevoordelende effecten van MTS bij scholieren met dyslexie	13
2.3.3 Aanpak voor ATS	15
2.4 De verschillende soorten ATS	17
2.5 Beschikbare tools en taalmodellen	22
2.6 De valkuilen bij AI en NLP	32
2.7 Conclusie	35
3 Methodologie	36
3.1 Requirementsanalyse	36
3.2 Vergelijkende studie	40
3.3 Prototype voor tekstvereenvoudiging	42
3.3.1 Tool voor leerkrachten	47
3.3.2 Tool voor scholieren	52
4 Resultaten	55
4.1 Requirementsanalyse	55
4.2 Vergelijkende studie	59
4.3 Opbouw van het prototype	64

5 Conclusie	66
6 Discussie	68
A Onderzoeksvoorstel	73
A.1 Introductie	73
A.2 State-of-the-art	75
A.2.1 Tekstvereenvoudiging	75
A.2.2 Noden van scholieren met dyslexie	75
A.2.3 Huidige toepassingen	77
A.2.4 Ontwikkelen met AI	77
A.3 Methodologie	78
A.4 Verwacht resultaat, conclusie	80
B Code voor tekstanalyse	81
C Code voor het prototype	85
Bibliografie	95

Lijst van figuren

1.1	Het leesplezier bij 15-jarigen volgens de PISA-test (De Meyer e.a., 2019).	2
2.1	Afbeelding uit Plavén-Sigraay e.a. (2017)	11
2.2	Afbeelding van Rello, Baeza-Yates, Dempere-Marco e.a. (2013)	13
2.3	Voorbeeld van PoS-labeling (Bilici, 2021).	16
2.4	Afbeelding van Althunayyan en Azmi (2021)	18
2.5	Afbeelding van Gooding 2022	26
2.6	Afbeelding van Gooding 2022	26
2.7	Afbeelding van Gooding 2022.	27
2.8	Experiment uit Binz en Schulz (2023) dat de mean-regret van de vier GPT-3 engines uittest.	29
2.9	Afbeelding uit McFarland (2023)	30
2.10	Afbeelding van Simon (2021)	31
2.11	Afbeelding van Ribas (2023)	32
3.1	Een mogelijke weergaven van de homepage.	45
3.2	Voorbeeldweergave van de instellingenpagina.	45
3.3	Stappenplan voor de ontwikkeling van het component voor lectoren.	50
3.4	Stappenplan voor de ontwikkeling van het component voor scholieren.	53
4.1	Resultaat Bing Chatbot	56
4.2	Illustratie van de tekstanalyse bij Simplish na een tekstvereenvoudiging.	57
4.3	Illustratie van de tekstanalyse bij Rewordify.	58
4.4	Schermafbeelding van SciSpace	59
4.5	Overzicht van het minimum, maximum en gemiddeld aantal woorden per zin per model in artikel 1.	60
4.6	Overzicht van het minimum, maximum en gemiddeld aantal woorden per zin per model in artikel 2.	61
4.7		61
4.8		62
4.9		62
4.10		63
A.1	(Readable, 2021)	79

Lijst van tabellen

2.1	Specifieke drempels bij het intensief lezen van een tekst.	7
2.2	Oplossingen die software-ontwikkelaars kunnen aanreiken bij een toe- passing of website.	8
2.3	Complexe leesfactoren van een wetenschappelijk artikel.	9
2.4	Prevalente leesgraadsscores.	10
2.5	Drie algemene technieken voor MTS	12
2.6	Bewezen voordelen van MTS op mensen met dyslexie.	15
2.7	Beschikbare Nederlandstalige, Engelstalige en meertalige lexicale da- tabanken anno mei 2023.	18
2.8	De drie manieren om extraherende samenvatting mogelijk te maken volgens Verma en Verma (2020).	20
2.9	Overzicht van gekende voorleessoftware, tekstvereenvoudigings- en samenvattingstools die intuïtief zijn ontwikkeld voor de eindgebruiker (leerkracht of scholier).	24
2.10	HuggingFace beschikbare en ge-finetuned taalmodellen	25
2.11	Tabel met alle GPT-3 parameters.	28
2.12	Samenvattend schema met vaak voorkomende struikelblokken bij NLP- toepassingen.	34
3.1	Shortlist van uit te testen tools en toepassingen voor tekstvereenvou- ding.	37
3.2	Criteria waarop toepassingen worden afgetoetst in de requirements- analyse	38
3.3	Het Moscow-schema, opgebouwd door middel van de requirements- analyse.	39
3.4	Gebruikte taalmodellen in de vergelijkende studie	40
3.5	De GPT-3-prompts die in de vergelijkende studie aan bod komen. . . .	42
3.6	Gebruikte programmeertalen in het prototype voor tekstvereenvoudi- ging.	43
3.7	Gebruikte Python-libraries en hun respectievelijke functie in het pro- totype.	44
3.8	Gebruikte SpaCy Word-embeddings	47
3.9	Alle beschikbare functionaliteiten in	48
3.10	Taken van de webontwikkelaar bij het uitwerken van het lerarencom- ponent.	51

3.11	Taken van de NLP Engineer bij het uitwerken van het lerarencomponent.	51
3.13	Taken van de system engineer bij het uitwerken van het lerarencomponent.	52
3.12	Taken van data engineer bij het uitwerken van het lerarencomponent.	52
3.14	Taken van NLP engineer bij het uitwerken van het scholierencomponent.	53
3.15	Taken van data engineer bij het uitwerken van het scholierencomponent.	54
4.1	Aantal zinnen	59

List of Listings

B.1	Script voor fase 1 van de vergelijkende studie.	81
B.2	Script voor de tweede fase van de vergelijkende studie.	82
B.3	Script voor fase 4 van de vergelijkende studie	83
C.1	Reader-klasse	85
C.2	HuggingFace-klasse	86
C.3	Reader-klasse	88
C.4	Writer-klasse omvattende de code om dynamische PDF- en Word-documenten te genereren.	91
C.5	Script voor het opstarten van de Docker-container voor Windows-gebruikers	93
C.6	Script voor het opstarten van de Docker-container voor Unix-gebruikers	94

1

Inleiding

Iedereen wordt dagelijks geconfronteerd met lezen. Deze vaardigheid strekt zich uit tot elk aspect van ons dagelijks leven. Dit geldt des te meer in het onderwijs, waar leraren worden aangemoedigd om diverse leesmaterialen te gebruiken om lesinhouden op een authentieke manier over te brengen. Wetenschappelijke artikelen kunnen ingezet worden als leesvoer voor scholieren in de derde graad van het middelbaar onderwijs, maar de leesgraad van deze artikelen brengt een nieuwe uitdaging voor zowel scholieren als leerkrachten met zich mee.

Zo stampte het Amerikaanse onderwijs C.R.E.A.T.E.¹ uit de grond. Dit initiatief zet scholieren tussen 12 en 18 jaar aan om wetenschappelijke artikelen te lezen in plaats van enkel boeken. Ze begrijpen hoe wetenschappers experimenten uitvoeren, plannen en resultaten analyseren en interpreteren. Vlaamse gelijkaardige initiatieven bestaan niet, maar Vlaamse lerarenopleidingen benadrukken het gebruik van divers didactisch leesmateriaal in de klas. Volgens het M-decreet en de leerplannen van het katholiek² en het gemeenschapsonderwijs³ worden Vlaamse leerkrachten geadviseerd om hun lessen op een toegankelijke manier aanbieden, zodat alle scholieren ongeacht leesachterstand worden meegenomen in het verhaal.

Vlaanderen is met een jaarlijks budget van 32 miljoen een pionier in Europa op het gebied van artificiële intelligentie (AI) op de werkvloer (Crevits, 2022). Zo stampte de Vlaamse overheid verschillende AI- projecten uit de grond, om Vlaamse AI ontwikkelingen te ondersteunen en om AI- softwarebedrijven te inspireren. Het amai-project⁴ brengt AI softwarebedrijven uit diverse domeinen samen, waaronder het

¹<https://teachcreate.org/>

²<https://pro.katholiekonderwijs.vlaanderen/basisoptie-stem/ondersteunend-materiaal>

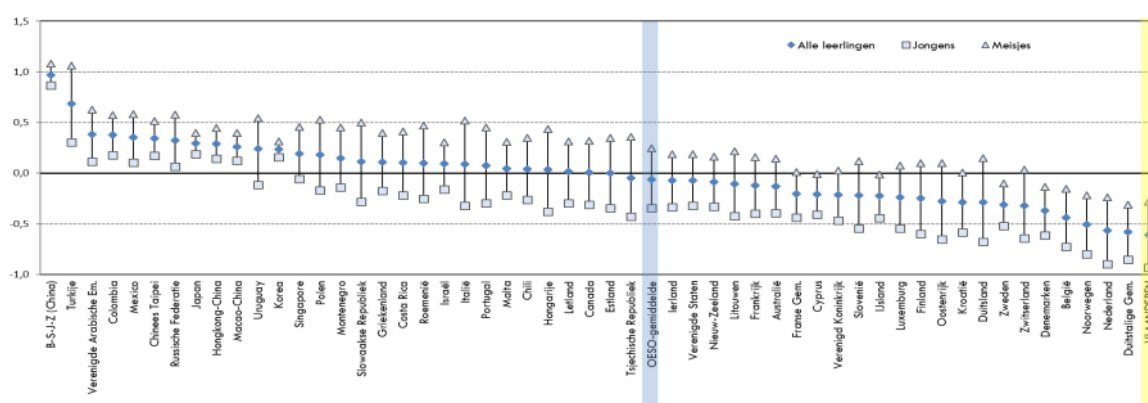
³<https://g-o.be/stem/>

⁴<https://amai.vlaanderen/>

onderwijs. Zij doelen op een automatisering van processen om de werkdruk bij leerkrachten te verminderen, door middel van real-time ondertiteling in de klas en een taalassistent voor leerkrachten in meertalige klasgroepen.

1.1. Probleemstelling

In 79 geïndustrialiseerde landen wordt de driejaarlijkse PISA-test afgenomen om de leesvaardigheid en wetenschappelijke geletterdheid van 15-jarige scholieren te meten. Uit de PISA-test van 2018 blijkt dat deze doelgroep in Vlaanderen zich echter negatief uit over leesplezier en daarmee de slechtst scorende doelgroep is van alle bevroagde landen. Zoals aangegeven in Figuur 1.1 beschouwt bijna de helft van de bevroagden intensief lezen als tijdverspilling en slechts 17% beschouwt lezen als een hobby. Dit is een dalende trend, want voordien lag deze trend hoger dan 20%. Boeiende topics uit vakliteratuur zoals Humo, Trends of EOS magazines kunnen belemmerd worden door deze vorm van media bij deze doelgroep.



Figuur (1.1)

Het leesplezier bij 15-jarigen volgens de PISA-test (De Meyer e.a., 2019).

Intensief leesbegrip valt niet te omzeilen in onze huidige samenleving, maar dit leesbegrip verschilt sterk in het middelbaar onderwijs. Zo benadrukt de inspectie van 2020 dat intensief lezen een essentiële vaardigheid is en een direct effect kan hebben op vakken buiten Nederlands, zoals bij het lezen van vraagstukken bij wiskundige vakken, of het begrijpen van vakjargon bij STEM-vakken.

Een doelgroep die extra met intensief leesbegrip geconfronteerd wordt, zijn scholieren met dyslexie. Onderzoeken van Bonte (2020) en van der Meer (2022) schatten dat ongeveer 15% van de Vlaamse scholieren in het middelbaar onderwijs een vorm van dyslexie heeft. Zo kunnen scholieren met dyslexie bij het intensief lezen geconfronteerd worden met een moeizame en stroeve automatisering bij het lezen en spellen. Hoewel scholieren met dyslexie ondersteuning kunnen krijgen,

mag de impact van leesstoornissen niet onderschat worden. De gevolgen hiervan kunnen zich namelijk doorzetten na het middelbaar onderwijs (Lissens e.a., 2020). Leesvaardigheid blijft daarmee cruciaal voor succes op school en in het werkveld. Scholieren met dyslexie kunnen problemen hebben met spelling, wat kan leiden tot onzekerheid en stress. Daarnaast zijn vooroordelen nog steeds een probleem en kunnen ze leiden tot stigmatisering. Echter toont onderzoek aan dat scholieren met dyslexie doorzettingsvermogen hebben en goede probleemoplossers zijn (Bonte, 2020; Ghesquière, 2018; Lissens e.a., 2020).

Het leerplan voor STEM-vakken stimuleert het gebruik van wetenschappelijke artikelen, maar houdt niet altijd rekening met de bijhorende complexe leesgraad. De ingewikkelde woordenschat en syntax in wetenschappelijke artikelen kunnen een hindernis vormen voor de begrijpelijkheid van een tekst, waardoor scholieren met dyslexie de kerninhoud moeilijk kunnen doorgronden. Het handmatig vereenvoudigen van wetenschappelijke artikelen kan planning, tijd en energie van leerkrachten in de derde graad middelbaar onderwijs opslorpen. Het Vlaamse middelbaar onderwijs staat onder druk en docenten hebben moeite om met deze werkdruk boven water te blijven.

Nu is AI technologisch voldoende hoogstaand om tekstvereenvoudiging te automatiseren en om een baanbrekende oplossing aan te bieden aan het middelbaar onderwijs. Soortgelijke technologieën worden echter amper toegepast in het onderwijs. Er is terughoudendheid door enerzijds ouders van leerlingen (Martens e.a., 2021b), anderzijds door de weinige ontwikkelingen in schoolgerelateerde AI-software. Dit terwijl AI-ondersteuning in het onderwijs wel degelijk een positief effect heeft (Belpaeme e.a., 2018; Kraft, 2020).

Recente technologieën bieden reeds mogelijkheden tot tekstvereenvoudiging aan, maar zijn voorlopig enkel in commandline of met scripts ter beschikking. Voor het gebruik van taalmodellen of API's is uitgebreide informaticakennis nodig, waarover de meeste scholieren en leraren niet beschikken. Anderzijds zijn de huidige online tools te beperkt en eerder gericht op samenvatten; wat niet noodzakelijk bijdraagt tot een eenvoudigere tekst. Er is nood aan een intuïtieve en gebruikersvriendelijke toepassing die taalmodellen of API's kan integreren en aanpassen naargelang de specifieke behoeften van een student met dyslexie. Zo kan dit enerzijds ook de werkdruk bij leerkrachten verminderen, anderzijds scholieren in de derde graad ondersteunen bij het lezen van complexe wetenschappelijke artikelen.

1.2. Onderzoeksvraag

Dit onderzoek beschrijft het gebruik van artificiële intelligentie in de vorm van tekstvereenvoudiging, als advies voor implementatie in het onderwijs. Specifiek om

scholieren met dyslexie in de derde graad van het middelbaar onderwijs te ondersteunen bij het intensief lezen van wetenschappelijke artikelen. Hiervoor is de volgende onderzoeksvraag opgesteld:

- Hoe kan een wetenschappelijke artikel automatisch vereenvoudigd worden, gericht op de unieke noden van scholieren met dyslexie in de derde graad middelbaar onderwijs?

Om deze onderzoeksvraag te kunnen beantwoorden, moet een antwoord gezocht worden op de volgende deelvragen:

1. Welke specifieke noden hebben scholieren met dyslexie van de derde graad middelbaar onderwijs bij het begrijpen van complexere teksten? Aanvullend hierop:
 - Wat zijn de specifieke kenmerken van wetenschappelijke artikelen?
2. Welke aanpakken zijn er voor tekstvereenvoudiging?
 - Hoe worden teksten handmatig vereenvoudigd voor scholieren met dyslexie?
 - Welke toepassingen, tools en modellen zijn er beschikbaar om Nederlandse geautomatiseerde tekstvereenvoudiging met AI mogelijk te maken?
 - Hoe kunnen geautomatiseerde tekstvereenvoudiging en gepersonaliseerde tekstvereenvoudiging gecombineerd worden?
3. Welke functies ontbreken AI-toepassingen om geautomatiseerde tekstvereenvoudiging mogelijk te maken voor scholieren met dyslexie in de derde graad middelbaar onderwijs?
 - Welke manuele methoden voor tekstvereenvoudiging ontbreken in deze tools?
4. Met welke valkuilen bij taalverwerking met AI moeten ontwikkelaars rekening houden?
5. Welke taalmodellen of LLM's zijn geschikt voor automatische tekstvereenvoudiging voor vereenvoudigde wetenschappelijke artikelen voor scholieren met dyslexie in de derde graad van het middelbaar onderwijs met dezelfde of gelijkaardige kwaliteiten als manuele tekstvereenvoudiging?
6. Hoe kan een intuïtieve lokale webtoepassing worden ontwikkeld die zowel scholieren met dyslexie als leerkrachten helpt bij het vereenvoudigen van wetenschappelijke artikelen met behoud van semantiek, jargon en zinsstructuren?

1.3. Onderzoeksdoelstelling

Het onderzoek heeft als doel om de technologische en logopedische aspecten te identificeren die AI-ontwikkelaars in overweging moeten nemen bij het creëren van een op maat gemaakte AI-toepassing voor geautomatiseerde tekstvereenvoudiging, specifiek ontwikkeld voor scholieren in de derde graad. Het resultaat van dit onderzoek is een prototype voor een AI-toepassing voor tekstvereenvoudiging in de vorm van een webtool. De webtool heeft twee functies. Enerzijds kan de tool de inhoud van wetenschappelijke artikelen vereenvoudigen op basis van de specifieke behoeften van scholieren met dyslexie in de derde graad van het middelbaar onderwijs. Anderzijds biedt de tool een geautomatiseerde benadering voor lectoren om wetenschappelijke artikelen te vereenvoudigen op basis van geselecteerde parameters en deze vervolgens in een bruikbaar formaat (pdf of word) terug te geven. De invoer bij dit prototype is een wetenschappelijk artikel in tekst- of PDF-formaat.

1.4. Opzet van deze bachelorproef

De rest van deze bachelorproef is als volgt opgebouwd:

In Hoofdstuk 2 wordt een overzicht gegeven van de stand van zaken binnen het onderzoeksdomein, op basis van een literatuurstudie.

In Hoofdstuk 3 wordt de methodologie toegelicht en worden de gebruikte onderzoekstechnieken besproken om een antwoord te kunnen formuleren op de onderzoeksvragen. Eerst wordt er een requirementsanalyse uitgevoerd, gevolgd door de ontwikkeling van een prototype voor tekstvereenvoudiging. In Hoofdstuk 4 worden de resultaten gegeven op dit onderzoek.

In Hoofdstuk 5, tenslotte, wordt de conclusie gegeven en een antwoord geformuleerd op de onderzoeksvragen. Ten slotte geeft Hoofdstuk 6 verdere aanbevelingen en aanzet voor toekomstig onderzoek binnen de bestudeerde domeinen.

2

Stand van zaken

2.1. Inleiding

Het onderzoek start met een uitgebreide literatuurstudie over de benodigde kennis binnen het logopedisch, taal- en informatica vakdomein om geautomatiseerde en gepersonaliseerde vereenvoudigde teksten te verkrijgen van wetenschappelijke artikelen. Om een toepassing voor gepersonaliseerde en geautomatiseerde tekstvereenvoudiging van wetenschappelijke artikelen op maat van deze doelgroep aan te reiken, is het van cruciaal belang om de noden van scholieren met dyslexie in de derde graad van het middelbaar onderwijs te benoemen. Het onderzoek benoemt bewezen noden met behulp van een literatuurstudie. Daarnaast moeten de problemen bij huidige wetenschappelijke artikelen ook aangekaart worden. Wetenschappelijke artikelen vereenvoudigen op maat voor scholieren met dyslexie kan volgens taalexperten op verschillende manieren. Het is belangrijk om stil te staan bij de bestaande en reeds bewezen handmatige tekstvereenvoudigingstechnieken. Vervolgens komen technieken voor geautomatiseerde tekstvereenvoudiging (ATV) aan bod. Om een beter begrip te hebben op deze technieken, wordt de nodige informatie van taalverwerking met AI gegeven, alsook huidige AI-technologieën die tekstvereenvoudiging kunnen realiseren. Ten slotte zijn AI-technologieën hoogstaand en worden alsmaar robuuster, maar het is cruciaal om bij dit onderzoek aandacht te besteden aan de mogelijke problemen die AI-ontwikkelaars moeten vermijden of waarvan zij zichzelf attent op moeten maken.

2.2. Specifieke noden en richtpunten

Om wetenschappelijke artikelen te vereenvoudigen op maat van de unieke noden van scholieren met dyslexie moet het onderzoek stilstaan bij de specifieke noden van scholieren met dyslexie in de derde graad van het middelbaar onderwijs, alsook bij de moeilijkheden bij het intensief lezen van wetenschappelijke artikelen.

Kenmerk	Bron
Trage woordbenoeming	(Bonte, 2020)
Problemen bij het leesbegrip	(Bonte, 2020; Gala & Ziegler, 2016)
Hardnekkig letter-voor-letter lezen	(Bonte, 2020; Zhang e.a., 2021)
Problemen met woordherkenning en -herinnering	(Bonte, 2020)
Moeite bij homofonische of pseudo-homofonische woordenschat	(Zhang e.a., 2021)
Moeite bij onregelmatige lettergreepcombinaties	Gala en Ziegler (2016)

Tabel 2.1: Specifieke drempels bij het intensief lezen van een tekst.

Deze sectie bespreekt eerst algemeen hoe scholieren met dyslexie bij het intensief lezen kunnen worden geholpen. Daarna worden de belemmeringen en moeilijkheden van wetenschappelijke artikelen aangekaart. Deze sectie beantwoordt de volgende twee onderzoeksvragen:

- Welke specifieke noden hebben scholieren met dyslexie van de derde graad middelbaar onderwijs bij het begrijpen van complexere teksten?
- Wat zijn de specifieke kenmerken van wetenschappelijke artikelen?

2.2.1. Specifieke noden van scholieren met dyslexie in de derde graad van het middelbaar onderwijs.

Leesvaardigheid is geen aangeboren vaardigheid, maar iets dat mensen zelf moeten aanleren (Bonte, 2020; van der Meer, 2022). Hoewel dit proces vlot kan verlopen, kunnen mensen met dyslexie benadeeld worden tijdens dit proces. Dyslexie wordt gekenmerkt door beperkt lezen en kan het voorlezen traag, radend en letter-voor-letter maken. Mensen met dyslexie kunnen tijdens het intensief lezen verschillende drempels ervaren; die worden in tabel 2.1 opgesomd.

[H]

De digitalisering evolueerde de voorbije twintig jaar in stijgende lijn en scholieren in de tweede en derde graad zijn door het gebruik van smartphones en laptops hier het meeste vatbaar voor, zowel voor thuisgebruik als binnen het onderwijs. Tieners tussen 15 en 18 zijn volgens 2023 digitaal geletterd, daarom speelt de digitale weergave een essentiële rol bij een dergelijke online toepassing. Verder omschrijft dit artikel een checklist van technische elementen waaraan een webpagina of toepassing moet voldoen om een leesbare ervaring te voorzien voor scholieren met

dyslexie. Deze noden worden weergegeven in tabel 2.2 en zijn gebaseerd op onderzoeken van 2023 en Santana e.a. (2012).

Zachtgele, -groene of -bruine achtergrondkleur	Consistente lay-out
Woord- en karakterspatiëring	Waarschuwingen geven omtrent formulieren, sessies (login)
Consistente lay-out	Duidelijk zichtbare koppen- of heading-structuur
Duidelijke symbolen gebruiken	Inhoud visueel groeperen
Huidige positie benadrukken	

Tabel 2.2: Oplossingen die software-ontwikkelaars kunnen aanreiken bij een toepassing of website.

2.2.2. Specifieke kenmerken van wetenschappelijke artikelen

Wetenschappelijke artikelen zijn van cruciaal belang voor het verspreiden van nieuwe kennis en onderzoeksresultaten, maar toch blijven ze voor velen een mysterieus en ontoegankelijk gebied vanwege de complexiteit van de inhoud en het technische jargon dat onmisbaar lijkt te zijn (Ball, 2017). Dit kan het begrip van de artikelen bemoeilijken, vooral bij intensief lezen, en vormt daarmee een extra obstakel bij het gebruik van wetenschappelijke artikelen als bron van kennis tijdens de les. Wetenschappelijke artikelen volgen IMRAD, een uniform formaat voor gepubliceerde wetenschappelijke artikelen, dat bestaat uit vijf hoofdstukken: samenvatting, inleiding, methodologie, resultaten en discussie. Hoewel deze artikelen als leermiddel worden gebruikt in het middelbaar en hoger onderwijs, is de inhoud van een hoger niveau en voornamelijk gericht op mensen uit het vakgebied waarvoor het artikel is geschreven. 2021 en Pain (2016) benadrukken de complexiteit van wetenschappelijke artikelen en de volgende aspecten waarom ze moeilijk te interpreteren zijn.

Deze factoren worden opgesomd in ???. Hoewel wetenschappelijke artikelen over een grote drempel bezitten, betrekken ze jongeren met wetenschappelijk onderzoek, alsook leren ze een discussieerbaar en kritische vaardigheid.

Probleem	Oplossing
Veel informatie in een compact formaat of <i>high information density</i>	Extra uitleg schrijven bij compacte zinnen, zoals extra uitleg bij woorden of zinnen herschrijven.
Hoog gebruik van meerlettergrepige woorden	Eenvoudigere synoniemen gebruiken.
Wetenschappelijk jargon	Rekening houden met een doelgroep buiten het vakgebied door eenvoudigere synoniemen te schrijven. Indien deze niet beschikbaar zijn, kan er extra uitleg als alternatief worden gegeven.
Complexe concepten	Paragrafen herschrijven zodat ze eerst uitleg geven op een high-level niveau. Vervolgens lagen van complexiteit toevoegen om de lezer te begeleiden doorheen de methodologie, discussie en conclusie van het wetenschappelijk artikel.
Cijfermateriaal bij resultaten	De interpretatie van percentages of cijfermateriaal schrijven. Zoals 'ongeveer een kwart van de bevolking' in plaats van '24.97% van de bevolking'.

Tabel 2.3: Complexe leesfactoren van een wetenschappelijk artikel.

Scholieren met verschillende achtergronden hebben vaak een verschillende achtergrondkennis, wat invloed kan hebben op het tekstbegrip. Bijvoorbeeld, scholieren met een achtergrond in fysica kunnen sneller de draad oppikken bij het lezen van fysica-gerelateerde artikelen dan scholieren met een economische achtergrond. Dit maakt het moeilijk om de leesbaarheid van een tekst objectief te beoordelen, omdat jargon voor de ene groep scholieren als gemakkelijk kan worden beschouwd, terwijl het voor de andere groep juist moeilijk kan zijn. Er is dus een verschil in behoeften tussen de verschillende groepen scholieren.

Er zijn door de jaren heen leesbaarheidsscores ontwikkeld om teksten objectief te kunnen beoordelen, aangezien ontoegankelijke en ingewikkelde zinsstructuren ook experts in het vakgebied kunnen hinderen. Er zijn twee bekende leesbaarheidsscores weergegeven in tabel ???. Dankzij Python-libraries, zoals Textstat¹ en

¹textstat

Readability², is het herschrijven van de formules niet meer nodig en bieden ze een snelle oplossing voor ontwikkelaars om een leesgraadscore te berekenen. Ook online toepassingen, zoals Textinspector³, kunnen analytisch inzicht geven in de complexiteit van Engelstalige teksten. Er bestaat momenteel geen alternatief voor Nederlandstalige wetenschappelijke artikelen. Het is wel belangrijk om te benadrukken dat deze leesbaarheidsscores geen rekening houden met de achtergrondkennis van mogelijke lezers.

Score	Uitleg
Flesch Reading Ease (FRE)	Deze leesbaarheidsscore wordt berekend op zinbasis. Hoe hoger de score, hoe 'eenvoudiger' de tekst.
Gunning FOG (FOG)	Deze leesbaarheidsscore wordt berekend op tekstbasis.

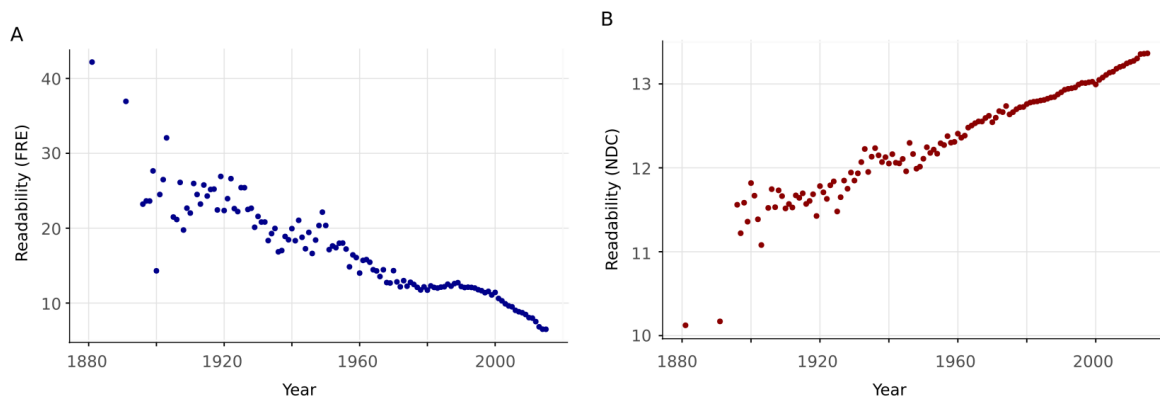
Tabel 2.4: Prevalente leesgraadsscores.

Diverse onderzoeken van de afgelopen tien jaar wijzen uit dat wetenschappelijke teksten steeds complexer worden, wat deze teksten voor niet-experten en niet-doctoraatsstudenten minder toegankelijk maakt vanwege het gebruik van technisch jargon en ingewikkelde zinsstructuren (Ball, 2017; Jones e.a., 2019; Plavén-Sigra e.a., 2017). Deze trend begon volgens onderzoek al in de tweede helft van de twintigste eeuw (Hayes, 1992).

Volgens onderzoek van Plavén-Sigra e.a. (2017) is de taal van wetenschappelijke artikelen de laatste jaren steeds complexer geworden. Uit een vergelijkende studie tussen abstracts en de rest van de inhoud van wetenschappelijke tijdschriften blijkt dat abstracts het meest complexe deel van een artikel vormen. De evolutie van de leesbaarheid wordt weergegeven in figuur 2.1, waarbij de FRE- (links) en NDC-scores (rechts) getoond worden. Het onderzoek schat dat op dit moment 22% van alle wetenschappelijke artikelen op het niveau van een masterstudent in het Engels geschreven zijn, tegenover 14% in 1960. Deze trend is belangrijk om op te volgen in het komende decennium, omdat het een obstakel kan vormen voor toekomstige generaties.

²<https://pypi.org/project/readability/>

³<https://textinspector.com/>

**Figuur (2.1)**

Afbeelding uit Plavén-Sigraay e.a. (2017)

Onbegrijpelijke en ontoegankelijke zinsstructuren hinderen ook vakexperten. Zo toonde onderzoek van McNutt (2014) aan dat begrip van de methodologie en resultaten cruciaal is in het kader van reproduceerbaarheid; enkel zo kunnen wetenschappers op correcte wijze een studie reproduceren en wetenschappelijke inzichten bevestigen of met verdere resultaten verrijken. Experimenten van Hubbard en Dunbar (2017) wijzen namelijk uit dat het net vooral de methodologie en resultaten van een wetenschappelijk artikel zijn die een hoge leesgraad vergen. In deze context is ook het onderzoek van Hartley (1999) en Snow (2010) relevant waarin ze aantonen dat het herschrijven van abstracts de begrijpbaarheid ervan kan verhogen.

Volgens Hollenkamp (2020) en McCombes (2022) moeten vereenvoudigde of samengevatte wetenschappelijke artikelen drie vragen kunnen beantwoorden: waarom werd het onderzoek uitgevoerd, wat zijn de experimenten en wat zijn de conclusies van de onderzoekers? Dit omvat de achtergrondinformatie, hypothesen, methoden, resultaten, implicaties, beperkingen en aanbevelingen. Om de tekst begrijpelijker te maken, kan deze worden omgezet in een ander formaat, zoals post-it notes, tabelvorm of opsommingen (Rijkhoff, 2022).

Wetenschappelijke artikelen zijn voornamelijk in PDF-formaat terug te vinden. Dit formaat valt eenvoudig in te lezen met verschillende Python-pakketten, waaronder PDFMiner of PyPDF. Wel kunnen ontwikkelaars problemen ondervinden bij het inlezen van PDF-bestanden, waarbij niet alle tekst of amper tekst uit de PDF kan worden geëxtraheerd. Als oplossing kunnen ontwikkelaars gebruik maken van *optical character recognition* of OCR. In Python zijn er pakketten die deze technologie eenvoudig kunnen realiseren, namelijk EasyOCR en Tesseract.

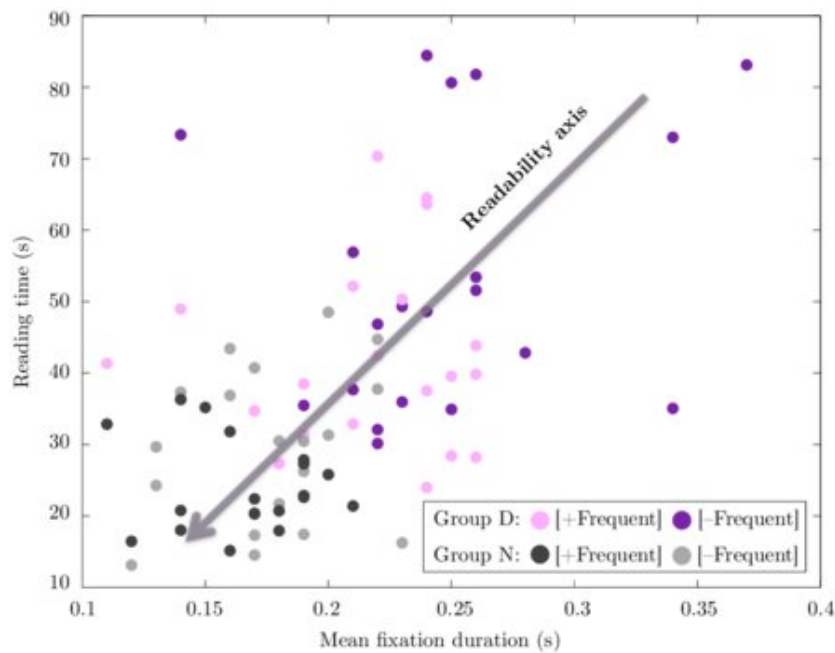
2.3. Aanpakken voor tekstvereenvoudiging

2.3.1. Manuele tekstvereenvoudiging

Voor sommige lezers kan het intensief lezen van een complexe tekst echter een uitdaging zijn, zoals scholieren met dyslexie. Om deze groep te helpen, wordt vaak *manual text simplification* toegepast (Siddharthan, 2014). *Manual text simplification* of MTS kan worden bereikt door het gebruik van eenvoudige woordenschat, zinsstructuren en formaatwijzigingen. MTS is het proces waarbij het technische leesniveau en het woordgebruik van een geschreven tekst wordt verminderd. Dit leidt tot een betere leeservaring zonder het verlies van de kerninhoud tijdens het lezen van de tekst.

Lexicale vereenvoudiging	Moeilijke woorden vervangen door eenvoudigere synoniemen Woorden- en synoniemenlijst maken Dubbelzinnige woorden vervangen Idiomen vervangen Rekening houden met het gekende jargon van de doelgroep
Syntactische vereenvoudiging	Tangconstructies aanpassen Zinnen langer dan negen woorden inkorten Verwijswoorden aanpassen Voorzetseluitdrukkingen aanpassen Samengestelde werkwoorden aanpassen Actieve stem gebruiken Onregelmatige werkwoorden vermijden
Formaatwijzigingen	Marges aanpassen Lettertype en -grootte aanpassen Woord- en karakterspatiëring aanpassen Herschrijven en -structureren als opsomming of tabelvorm

Tabel 2.5: Drie algemene technieken voor MTS

**Figuur (2.2)**

Afbeelding van Rello, Baeza-Yates, Dempere-Marco e.a. (2013)

2.3.2. Bevoordelende effecten van MTS bij scholieren met dyslexie

Onderzoek toont aan dat vereenvoudigde teksten het leesbegrip en woordherkenning van kinderen met dyslexie significant kunnen verbeteren (Rivero-Contreras e.a., 2021). Bovendien blijkt uit experimenten dat frequent woordgebruik de ontcijfertijd bij mensen met dyslexie significant vermindert, en dat teksten met verminderde lexicale complexiteit minder leesfouten opleveren voor mensen met dyslexie (Gala & Ziegler, 2016; Rello, Baeza-Yates, Dempere-Marco e.a., 2013). Deze studies benadrukken ook moeilijkheden van kinderen met dyslexie bij het lezen van woorden met onregelmatige lettergreepcombinaties. Mensen zonder dyslexie bereiken doelwaarden onder optimale omstandigheden, zoals aangegeven door de richting van de pijl op figuur 2.2. Het gebruik van veelvoorkomende woorden vermindert de decodeertijd en verbetert het leesbegrip voor mensen met dyslexie.

Hoewel onderzoeken de positieve effecten van lexicale vereenvoudiging voor lezers met dyslexie onderstrepen, is er relatief weinig onderzoek gedaan naar de effecten van syntactische vereenvoudiging op kinderen en scholieren met dyslexie. In het experiment van Linderholm e.a. (2000) had het aanpassen van causale structuren een significant effect op het leesbegrip en de foutenmarge van de bevroagden met een lage leesgraad. Door coherentieonderbrekingen te herstellen en tekstgebeurtenissen in een tijdsafhankelijke volgorde te plaatsen, konden zowel vaardige als minder vaardige lezers profiteren van de revisies. Verbaal parafraseren had geen significant effect op lezers met dyslexie, volgens Rello, Baeza-Yates en

Saggion (2013). De bevroagden waren tussen de 13 en 37 jaar oud, met een gemiddelde leeftijd van 21 jaar. Het tekstformaat bleef ongewijzigd, maar lettertypes werden aangepast.

Een andere vorm van aanpassingen waarmee geëxperimenteerd werd om de leesbaarheid van teksten voor de doelgroep te verhogen, is het personaliseren van samenvattingen (Nandhini & Balasundaram, 2013). Het experiment in het onderzoek maakt gebruik van onaangepaste zinnen uit de oorspronkelijke tekst die op maat van de lezer zijn gepresenteerd en herstructureert deze volgens de oorspronkelijke tekst. Door de belangrijkste zinnen onaangepast te laten en de structuur aan te passen, wordt de tekst toegankelijker gemaakt voor de lezer. Hoewel de resulterende logische structuur door de onderzoekers in twijfel werd getrokken, was de leesbaarheid van teksten bij de deelnemers significant beter dan bij de oorspronkelijke tekst, zonder negatieve effecten op het leesbegrip.

Tot slot hebben onderzoeken aangetoond dat scholieren met dyslexie gevoeliger zijn voor veranderingen in visuele parameters, zoals lettertype, karakterafstand, tekst- en achtergrondkleur en grijswaarden. Om de leesbaarheid te verbeteren, worden minimalistische ontwerpen met pictogrammen en afbeeldingen aanbevolen, evenals lettergrootte groter dan 14pt en een sans-serif, monospaced of roman lettertype. Volgens Bezem en Lugthart (2016), Rello en Baeza-Yates (2015) en Rello en Bigham (2017) zijn lichtgrijze achtergronden met zwart lettertype op een gele achtergrond, of zachtgele, -groene of lichtblauwe achtergrondkleuren de beste kleurencombinaties. Het gebruik van lettertypen zoals OpenDys heeft geen effect op lezers met of zonder dyslexie, terwijl cursieve lettertypen worden afgeraden, aldus Rello en Baeza-Yates (2013) en Rello en Baeza-Yates (2015).

In tabel 2.6 worden de bewezen strategieën opgesomd met hun samenhangend voordeel.

Techniek	Bewezen voordeel
Frequent woordgebruik	Lagere decodeertijd Beter leesbegrip
Verwerpen van onregelmatige lettergrepen	Verminderde decodeertijd Beter leesbegrip
Causale structuren aanpassen	Beter leesbegrip Minder fouten bij het intensief lezen
Tekstgebeurtenissen in een tijdsafhankelijke volgorde plaatsen	Beter leesbegrip bij het reviseren
Coherentieonderbrekingen herstellen	Beter leesbegrip bij het reviseren
Gepersonaliseerde samenvatting	Betere leesbaarheid
Zachtkleurige achtergrond	Betere leesbaarheid
Niet-cursieve, sans-serif lettertypen	Betere leesbaarheid
Lettertype groter dan 14pt	Betere leesbaarheid

Tabel 2.6: Bewezen voordelen van MTS op mensen met dyslexie.

2.3.3. Aanpak voor ATS.

Met de laatste evolutie in machinaal leren is dit een proces dat geautomatiseerd kan worden. *Automatic text simplification* of ATS is een onderdeel van natuurlijke taalverwerking binnen machinaal leren (ML). Dit omvat verschillende technieken zoals tekstanalyse, taalherkenning, -generatie, spraakherkenning en -synthese, en semantische analyse. NLP stelt computers in staat om menselijke taal te begrijpen en te communiceren op een natuurlijke manier.

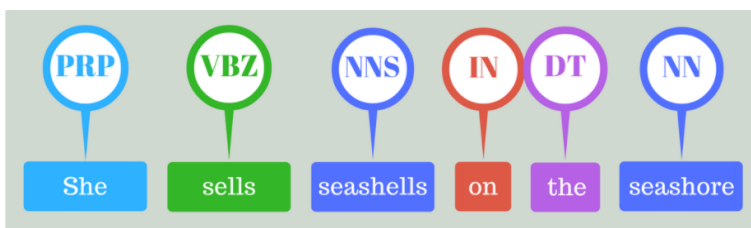
De begrippen die volgen worden behandeld in Eisenstein (2019) en Sohom (2019) en zijn cruciaal voor de daaropvolgende concepten.

Tokenisatie splitst zinnen op basis van tokens. Er zijn vier manieren om tokens in een tekst te splitsen en zo een woordenschat op te bouwen, namelijk op woord-, karakter-, subwoord- en zinniveau, volgens onderzoek van Menzli (2023). Bij karakter-

tokenisatie neemt de inputlengte toe, maar dit heeft volgens Ribeiro e.a. (2018) weinig bruikbare use cases. Het opsplitsen van zeldzame woorden in kleinere stukken om een woordenschat op te bouwen biedt voordelen ten opzichte van tokenisatie op woordniveau, aldus (Iredale, 2022).

In NLP is het lemmatiseren gebaseerd op stemming, maar het houdt ook rekening met de betekenis van elk woord. Er zijn Nederlandstalige modellen beschikbaar voor lemmatiseren, zoals JohnSnow⁴. Bij omgekeerd lemmatiseren wordt een afgeleide van de stam bepaald, bijvoorbeeld enkelvoud of meervoud voor zelfstandige naamwoorden zoals 'hond' (Eisenstein, 2019). Bij het parsen krijgt elk woord of zinsdeel een label toegewezen, zoals zelfstandig naamwoord, bijwoord, werkwoord, bijzin of stopwoord. Het identificeren van zinsdelen wordt 'chunking' genoemd. Parsing heeft een ambiguïteitsprobleem omdat bijvoorbeeld een 'plant' niet gelijk is aan de vervoeging van het werkwoord 'planten' (Eisenstein, 2019).

Om de betekenis van elk woord in een tekst te begrijpen, moet een machine in staat zijn om de betekenis achter elk token te begrijpen. Dit is waar *sequence labeling* om de hoek komt kijken, volgens Eisenstein (2019). Elk woord in een tekst wordt gekoppeld aan een label voor *Part-of-Speech* (PoS) of *Named Entity Recognition* (NER). Deze fase van NLP achterhaalt de structuur van een tekst. PoS-tagging richt zich op de grammaticale categorieën van woorden, terwijl NER-labeling zich richt op het herkennen van specifieke entiteiten in een tekst. Bij PoS-tagging worden de woorden in een zin geanalyseerd. Elk woord wordt gekoppeld aan een grammaticale categorie, zoals een zelfstandig naamwoord, werkwoord, bijvoeglijk naamwoord of bijwoord. *PoS-tagging* helpt bij het begrijpen van de syntactische structuur van een zin en is nuttig bij parsing en machinevertaling. Een voorbeeld van PoS-tagging is te zien in figuur 2.3 en is afkomstig uit Bilici (2021).



Figuur (2.3)

Voorbeeld van PoS-labeling (Bilici, 2021).

⁴https://nlp.johnsnowlabs.com/2020/05/03/lemma_nl.html

NER-labeling wordt gebruikt om namen van personen, organisaties en locaties te herkennen en te classificeren. Het is een techniek om specifieke informatie uit tekst te halen, zoals het identificeren van de namen van personen, plaatsen of bedrijven die in nieuwsartikelen worden genoemd, of het extraheren van belangrijke data of getallen uit financiële rapporten, aldus Jurafsky e.a. (2014). Figuur ?? illustreert deze taak. Er zijn vier verschillende vormen van NER-labeling, zoals beschreven door J. Li e.a. (2018): *dictionary-based*, *rule-based*, *ML-based* en *deep learning-based*. De eerste twee maken gebruik van vooraf gedefinieerde woordenboeken en regels, terwijl de laatste twee gebruik maken van statistische of neurale netwerken om te leren hoe entiteiten te herkennen. Elke vorm maakt gebruik van verschillende kenmerken en representaties om entiteiten te modelleren. Poel e.a. (2008) heeft een neurale netwerkmodel onderzocht voor PoS-tagging van Nederlandstalige teksten. Het model behaalde een nauwkeurigheid van 97,88% voor bekende woorden en 41,67% voor onbekende woorden en maakte gebruik van de Corpus Gesproken Nederlands (CGN) als trainingsdata. In de verwerking van tekst maken NLP-systemen gebruik van embeddings om woorden numeriek te representeren. Traditionele word embeddings bouwen een woordenschat op zonder de betekenis ervan in context te begrijpen. Contextuele word embeddings begrijpen daarentegen wel de context van een woord (Eisenstein, 2019).

2.4. De verschillende soorten ATS

Tekstvereenvoudiging kan bijdragen aan het begrijpen van complexe informatie. Zoals onderzocht door Siddharthan (2014), zijn er vier soorten transformaties bij geautomatiseerde tekstvereenvoudiging, waaronder lexicale vereenvoudiging, waarbij complexe woorden worden vervangen door eenvoudiger synoniemen. Bij *lexical simplification* (LS) of lexicale vereenvoudiging worden ingewikkelde woorden vervangen door eenvoudiger synoniemen. Bijvoorbeeld, 'adhesief' kan vervangen worden door 'klevend'. Kandula e.a. (2010) noemt twee manieren om lexicale vereenvoudiging te bewerkstelligen: het vervangen van het woord door een synoniem of het genereren van extra uitleg. De zinsstructuur blijft hetzelfde en de kerninhoud en benadrukking van de tekst blijven behouden. Het doel van lexicale vereenvoudiging is om de moeilijkheidsgraad van de woordenschat in een zin of tekst te verlagen.

Verschillende onderzoeken hebben aangetoond dat lexicale vereenvoudiging een belangrijke bijdrage kan leveren aan het begrijpen van complexe informatie, en in dit kader wordt de pipeline zoals weergegeven in figuur 2.4 vaak gebruikt, bijvoorbeeld in onderzoeken van Bingel e.a. (2018), Bulté e.a. (2018) en Paetzold en Specia (2016). Deze pipeline omvat bij de vermelde onderzoeken telkens minstens vier stappen, waarbij de eerste stap *Complex Word Identification* (CWI) is, een gesuperviseerde NLP-taak die moeilijke woorden of *multi-word expressions* (MWE) in

een tekst identificeert (Gooding & Kochmar, 2019; Shardlow, 2013). Na CWI kan lexicale vereenvoudiging (LS) worden toegepast, waarbij complexe woorden worden vervangen door eenvoudigere synoniemen of verder worden uitgelegd met voorbeelden of definities (Kandula e.a., 2010; Zeng e.a., 2005). Het is van cruciaal belang dat CWI goed wordt uitgevoerd, omdat een lage *recall* van dit component zal resulteren in een uitvoertekst waarin moeilijke woorden niet worden vereenvoudigd, zoals opgemerkt door Shardlow (2013). Er zijn verschillende manieren geïdentificeerd om substitutiegeneratie uit te voeren, zoals opgesomd in tabel 2.7. Recenter onderzoek, zoals dat van Zhou e.a. (2019), gebruikt ook een extra *Substitution Ranking* (SR) stap om substituties te rangschikken op basis van relevantie.



Figuur (2.4)

Afbeelding van Althunayyan en Azmi (2021)

Databank	Ondersteunde talen
Engels	WordNet SWORDS LSBert
Nederlands	Celex NT2Lex Cornetto
Meertalig (Engels, Duits, Spaans en Portugees)	PHOR-in-One

Tabel 2.7: Beschikbare Nederlandstalige, Engelstalige en meertalige lexicale databanken anno mei 2023.

Om de complexiteit van een tekst te verminderen, kan syntactische vereenvoudiging worden toegepast door de grammatica en zinsstructuur aan te passen. Dit kan gedaan worden door het combineren van twee zinnen tot één eenvoudigere zin of door de syntax te vereenvoudigen, terwijl de inhoud behouden blijft. Een voorbeeld van zo'n model is ontwikkeld door Kandula e.a. (2010) voor medische informatie. Het model bestaat uit drie modules, die zinnen met meer dan tien woorden vereenvoudigen en eventueel vervangen door kortere zinnen. Het omvat een

PoS Tagger, een Grammar Simplifier en een Output Validator als onderdelen van de architectuur.

1. Ten eerste wordt de PoS Tagger-functie uit het open-source pakket OpenNLP⁵ gebruikt.
2. Vervolgens splitst de *Grammar Simplifier*-module lange zinnen in kortere zinnen door middel van het identificeren van POS-patronen en het toepassen van transformatieregels.
3. Tot slot controleert de *Output Validator*-module de grammatica en leesbaarheid van de output van de Grammar Simplifier.

Geautomatiseerde tekstvereenvoudiging is geen nieuw concept. Volgens onderzoeken van Canning e.a. (2000) en Siddharthan (2006) waren de eerste aanpakken op geautomatiseerde tekstvereenvoudiging gebouwd op rule-based modellen. Deze modellen bewerken de syntax door zinnen te splitsen, te verwijderen of de volgorde van de zinnen in een tekst aan te passen. Lexicale vereenvoudiging kwam hier niet aan de pas. Enkel bij recentere onderzoeken van Bulté e.a. (2018) en Coster en Kauchak (2011) werd het duidelijk hoe lexicale en syntactische vereenvoudiging gecombineerd kon worden.

Vroegere onderzoeken tonen aan dat geautomatiseerde tekstvereenvoudiging al geruime tijd bestaat. Zo hebben Canning e.a. (2000) en Siddharthan (2006) onderzocht dat de eerste methoden gebaseerd waren op *rule-based* modellen die de syntaxis van de tekst bewerken door zinnen te splitsen, te verwijderen of te herschikken. Lexicale vereenvoudiging speelde hierbij geen rol. Pas bij recentere onderzoeken, zoals die van Coster en Kauchak (2011) en Bulté e.a. (2018), werd het duidelijk hoe lexicale en syntactische vereenvoudiging gecombineerd konden worden.

Om wetenschappelijke artikelen toegankelijker en begrijpelijker te maken, is het van belang om de kernpunten van een artikel op een duidelijke en beknopte manier samen te vatten. Hoewel samenvatten niet gericht is op het vereenvoudigen van de tekst, is het wel een techniek die noodzakelijk is om de semantiek achter een tekst met zo min mogelijk woord- of tekens te kunnen begrijpen. Full-text-search en gepersonaliseerde informatiefiltering benadrukken het belang van deze op maat gemaakte samenvattingen. Een samenvattingssysteem bestaat uit drie fases: analyse van de brontekst, identificatie van de kernpunten en samenvoeging van deze kernpunten tot één overzichtelijke tekst. Het machinaal samenvatten van

⁵<https://opennlp.apache.org/>

teksten is geen nieuw concept en kan op twee manieren worden gedaan: door extractie en door abstractie (DuBay, 2004; Hahn & Mani, 2000).

Bij het extraherend samenvatten van teksten worden de belangrijkste zinnen gemarkeerd en opnieuw geformuleerd. Dit kan echter leiden tot onsamenvattende uitvoertekst, zoals Khan (2014) heeft aangetoond. Om de kernzinnen te bepalen, kunnen verschillende methoden worden gebruikt, zoals woordfrequentie, zinpositie -en gelijkenissen, de cue-methode, titels, de rest van het document, proper nouns, woordgebruik en de afstand tussen text units met entiteiten, aldus Khan (2014). Verschillende technieken voor het extraherend samenvatten van teksten, waaronder graafgebaseerde methoden, maximal marginal relevance (MMR) en metaheuristisch gebaseerde ES, zijn onderzocht door Verma en Verma (2020) en worden verder beschreven in tabel ??.

MMR-gebaseerde ES	Deze traditionele techniek gebruikt de maximaal marginale relevantiescore (MMR) om de relevantie en diversiteit van gemarkeerde zinnen te bepalen. Zorgt ervoor dat de geselecteerde zinnen niet te veel overlappen in inhoud en relevantie. Kan leiden tot betere samenvattingen, maar vereist meer rekenkracht en tijd.
Graafgebaseerde ES	Deze techniek vertegenwoordigt een document als een graaf van zinnen en gebruikt algoritmen om de belangrijkste zinnen te bepalen en redundantie te vermijden. Kan zowel voor lange wetenschappelijke artikelen als korte nieuwsartikelen goede resultaten opleveren (Lin & Bilmes, 2010; McDonald, 2007).
Metaheuristisch-gebaseerde ES	Deze techniek maakt gebruik van optimalisatie-algoritmen zoals genetische algoritmen en zwermoptimalisatie om de belangrijkste zinnen in een tekst te vinden (Premjith e.a., 2015; Verma & Verma, 2020). Evaluatiefunctie kan in een lokaal optimum vastlopen afhankelijk van de gebruikte criteria. (Rani & Kaur, 2021).

Tabel 2.8: De drie manieren om extraherende samenvatting mogelijk te maken volgens Verma en Verma (2020).

De extraherende samenvatting van nieuwsartikelen kan beïnvloed worden door de vooroordelen of *bias* van de auteur, zo blijkt uit experimenten uitgevoerd door McKeown e.a. (1999). Bij deze vorm van samenvatten worden de zinnen overgenomen zoals ze zijn. Hahn en Mani (2000) bouwde verder op deze experimenten door het

combineren van *knowledge-rich* en *knowledge-poor* methoden, wat resulteerde in significante verbeteringen. Bij het extraherend samenvatten is het van belang om de meest relevante tekstgedeeltes te selecteren, meestal in de vorm van zinnen. Om de lexicale en statistische relevantie van een zin te kunnen bepalen, worden twee methoden genoemd door Hahn en Mani (2000):

- Het lineaire gewichtsmodel, waarbij elke teksteenheid wordt gewogen op basis van factoren zoals de positie van de zin en het aantal keren dat deze voorkomt.
- Het gewichtsmodel op basis van de statistische relevantie van een eenheid, waarbij rekening wordt gehouden met de aanwezigheid van woorden in titels.

Om de nauwkeurigheid van modellen te verbeteren, hebben Nallapati e.a. (2017) *SummaRuNNer* ontwikkeld: een oplossing voor het extraherend samenvatten van teksten met behulp van een neuraal netwerk. Het systeem is opgebouwd met *PyTorch* en bestaat uit drie modellen: een recurrent neuraal netwerk, een convolutieel recurrent neuraal netwerk en een hiërarchisch attention network.

Extraherende samenvattingen kunnen leiden tot een onsamenhangende tekst. Een oplossing hiervoor is abstraherende samenvatting, die vergelijkbaar is met parafraseren. Er zijn twee benaderingen om een tekst op een abstraherende manier samen te vatten: de semantisch-gebaseerde en de structuurgebaseerde benadering. De structuurgebaseerde methode gebruikt regels om belangrijke informatie in de tekst te vinden, maar dit kan leiden tot samengevatte zinnen van lage taalkundige kwaliteit en grammaticale fouten. De semantisch-gebaseerde benadering daarentegen gebruikt de betekenis van de tekst om korte en duidelijke samenvattingen te maken met minder redundante zinnen en een betere taalkundige kwaliteit, hoewel er mogelijk een extra parsingfase nodig is. Onderzoek door Cao (2022) richt zich op het gebruik van deep learning-methoden om automatisch abstraherende samenvattingen te genereren. Modellen zoals RNN's, CNN's en Seq2Seq kunnen worden gebruikt voor abstraherende samenvattingen door de betekenis van de tekst te begrijpen en belangrijke informatie over te brengen (Suleiman & Awajan, 2020).

Wanneer het aankomt op het samenvatten van lange tekstdocumenten zoals boeken of wetenschappelijke tijdschriften, is een andere aanpak vereist vanwege de lengte van het oorspronkelijke document. Het opsplitsen van de tekst kan leiden tot het breken van samenhangende paragrafen, wat later kan resulteren in redundante tekst in het samengevatte document. Volgens onderzoek van zowel Hsu e.a. (2018) als Huang e.a. (2019) zou een ideale samenvatting van een lang document zowel extraherend als abstraherend moeten zijn. Om deze reden zou een hybride

samenvattingspijplijn twee fasen moeten bevatten: een inhoudselectiefase waarin de kernzinnen worden geëxtraheerd door middel van extraherende samenvatting en een parafraserende fase waarin de geïdentificeerde kernzinnen abstraherend worden samengevat.

Bij het ontwerpen van een ondersteunende toepassing voor dyslexie moet rekening worden gehouden met de individuele behoeften en uitdagingen van elke scholier, volgens Gooding (2022). Dyslexie kan zich namelijk op verschillende manieren uiten bij verschillende scholieren, waarbij bijkomende symptomen niet altijd van invloed zijn op de spellingprestaties van een scholier. Om deze reden is het belangrijk om een toepassing te ontwerpen die rekening houdt met de diversiteit van dyslexie.

2.5. Beschikbare tools en taalmodellen

Het kan moeilijk zijn voor scholieren met dyslexie om goed te lezen en te schrijven. Gelukkig zijn er verschillende softwareprogramma's en tools beschikbaar om hen te ondersteunen. Deze sectie gaat in op de nationale en internationale software die specifiek is ontworpen voor scholieren met dyslexie om hen te helpen bij het lezen van teksten. Er zal voornamelijk worden gekeken naar beschikbare software in Vlaamse middelbare scholen, chatbots zoals Bing AI en ChatGPT, en software die speciaal is ontwikkeld om dyslexie bij het lezen te ondersteunen. Deze sectie beantwoordt de volgende deelvraag:

- Welke toepassingen, tools en modellen zijn er beschikbaar om Nederlandstalige geautomatiseerde tekstvereenvoudiging met AI mogelijk te maken?

Scholieren met dyslexie krijgen in het middelbaar onderwijs enkel ondersteuning in de vorm van voorleessoftware (De Craemer e.a., 2018; OnderwijsVlaanderen, 2023). Het ministerie van Onderwijs in Vlaanderen biedt licenties aan voor verschillende softwarepakketten zoals SprintPlus, Kurzweil3000, Alinea Suite, IntoWords en TextAid, die scholieren kunnen gebruiken om zinnen te markeren en deze vervolgens samen te vatten. Het samenvatten gebeurt echter op een manier waarbij de zinnen lexicaal en syntactisch identiek blijven. Helaas bieden deze softwarepakketten geen functie voor het vereenvoudigen van teksten. Volgens Tops e.a. (2018) is het belangrijk om deze software zo vroeg mogelijk in de schoolcarrière te introduceren, zodat scholieren er snel vertrouwd mee raken en het optimaal kunnen gebruiken in verdere studies. Hoewel Tops e.a. (2018) de handige aspecten van deze software benadrukt, is het te laat om deze software pas in het hoger onderwijs te introduceren.

Momenteel hebben scholieren met dyslexie in het middelbaar onderwijs beperkte tekstvereenvoudigingsfunctionaliteiten in de bestaande voorleessoftware. Dit onderstreept de noodzaak aan nieuwe erkende tools die tekstvereenvoudiging in het onderwijs mogelijk maken. Tools zoals Simplish en Rewordify kunnen een oplossing bieden. Hoewel Simplish oorspronkelijk Engelstalig is, is het inmiddels mogelijk om een vereenvoudigde tekst te genereren van Nederlandstalige teksten, al is deze functionaliteit betalend voor niet-Engelstalige teksten. Rewordify is enkel in staat om Engelstalige teksten te vereenvoudigen. Er zijn echter maar weinig proof-of-concepts beschikbaar en de taalmodellen waarop deze applicaties werken zijn niet gekend om deze tools te kunnen repliceren. Bovendien zijn er geen API's beschikbaar om mee te werken.

Voor samenvatting zijn er echter meer tools beschikbaar. Enkele voorbeelden hiervan zijn Resoomer, Paraphraser, Editpad, Scribbr en Quillbot. Al zijn er onderzoeken over geautomatiseerde lexicale en syntactische vereenvoudiging voor scholieren met dyslexie, het aantal onderzoeken over samenvatten voor deze doelgroep is schaars. Zoals eerder aangehaald is er wel onderzoek gedaan naar de verschillende manieren om een tekst samen te vatten, maar er is geen toepassing of onderzoek dat dit concreet uitwerkt. Stajner (2021) wijzen erop dat toepassingen voor tekstvereenvoudiging regelmatig als *showcase* van de technologie ontwikkeld worden en zelden tot weinig rekening houden met gepersonaliseerde samenvatting om zo rekening te houden met de verschillende noden.

Er zijn weinig toepassingen beschikbaar om wetenschappelijke artikelen te vereenvoudigen, maar er bestaan enkele gratis en betalende opties, waaronder SciSpace⁶ en Scholarcy⁷. Hoewel het proof-of-concept voor een webtoepassing genaamd Lexi, nu genaamd Hero, oorspronkelijk was ontworpen om teksten te vereenvoudigen voor mensen met dyslexie (Bingel e.a., 2018), is het niet geschikt voor wetenschappelijke artikelen. Het is echter nu herwerkt als browserextensie voor selecte nieuwssites. Deze tools worden samengevat in tabel 2.9.

⁶<https://typeset.io/>

⁷<https://www.scholarcy.com/>

Tool	Algemene functionaliteit
Sprintplus Kurzweil3000 Alinea Suite IntoWords TextAid	Voorleessoftware met ondersteuningsmogelijkheden voor woordenschat
Resoomer Paraphraser Editpad Scribbr Quillbot	Samenvattingstool zonder specifieke casus
SciSpace Scholarcy	Samenvattingstool voor wetenschappelijke artikelen.
Simplish Rewordify	Tool voor tekstvereenvoudiging met bijhorende analyse

Tabel 2.9: Overzicht van gekende voorleessoftware, tekstvereenvoudigings- en samenvattingstools die intuïtief zijn ontwikkeld voor de eindgebruiker (leerkracht of scholier).

Huggingface wordt beschouwd als een platform of portaalsite voor het delen van ML-modellen en datasets. Het biedt een breed scala aan API's en tools die gemakkelijk te downloaden en trainen zijn voor pretrained modellen voor veelvoorkomende NLP-taken, zoals tekstclassificatie, taalmodellering en samenvatting. Ontwikkelaars kunnen deze modellen finetunen op specifieke datasets om snel modellen te bouwen en inzetten voor vereenvoudigings- en samenvattingstaken. Bruikbare taalmodellen met hun respectievelijke casus worden weergegeven in tabel 2.10.

Taalmodel	Specifieke casus
Google Pegasus	Samenvattingstaken voor kort tot middelgrote documenten.
Longformer Encoder-Decoder	Samenvatting van lange wetenschappelijke artikelen
Haining Scientific Abstract Simplifica- tion ⁸	Het lexicaal vereenvoudigen van wetenschappelijke artikelen.
BART Large Scienti- fic Summarisation ⁹	Het samenvatten van wetenschappelijke artikelen.
T5 finetuned text simplification mo- del ¹⁰	Tekstvereenvoudiging voor algemeen gebruik.
Keep It Simple ¹¹	Ongesuperviseerde tekstvereenvoudiging met ATS.

Tabel 2.10: HuggingFace beschikbare en ge-finetuned taalmodellen

Het taalmodel GPT-3 is ontwikkeld door OpenAI en maakt gebruik van een tweestapsleerparadigma. Het model wordt eerst ongesuperviseerd getraind met een taalmodelleringsdoel en wordt vervolgens gesuperviseerd gefinetuned. Het aantal parameters van het model is sterk toegenomen van 1,5 miljard bij GPT-2 naar 175 miljard bij GPT-3, en het model is getraind op niet-gecategoriseerde data van het internet met behulp van datasets zoals Common Crawl, WebText2, Books1, Books2 en Wikipedia (C. Li, 2022; Radford e.a., 2019). GPT-3 heeft verschillende versies, waaronder GPT-3.5 die als engine dient voor ChatGPT, een conversatie-chatbot. Omdat onbegrijpelijke en ontoegankelijke zinsstructuren niet alleen voor leken, maar ook voor vakexperten een obstakel vormen, is het belangrijk om te benadrukken dat GPT-3.5 gericht is op conversationele doeleinden, terwijl GPT-3 in het algemeen bedoeld is om met hoogstens één prompt te werken (Hubbard & Dunbar, 2017; McNutt, 2014).

Volgens Gooding (2022) kan een toepassing zoals ChatGPT de leesbaarheid van een tekst op twee manieren bevorderen. Ten eerste kan het verschillende versies van een tekstvereenvoudiging leveren, waardoor de gebruiker een keuze kan maken uit verschillende mogelijke vereenvoudigingen. Dit wordt geïllustreerd in figuur 2.6, waar Gooding aangeeft dat de invoertekst op verschillende manieren kan worden vereenvoudigd op basis van de prompt. Verder bespreekt Gooding in figuur 2.7 de prompt-gebaseerde evaluatiemanier van GPT-3. Het model kan de verschillen tussen de gegenereerde teksten begrijpen en de gebruiker feedback geven. In

figuur 2.5 laat Gooding zien hoe het model lexicale, conceptuele en syntactische vereenvoudiging kan toepassen op een paragraaf uit een boek van de Russische schrijver Dostoevsky.



Can you simplify the following paragraph:

At some ideas you stand perplexed, especially at the sight of sin. And you must ask yourself whether to combat it with force or with humble love. Always decide to combat it with humble love. If you decide this once and for all, you can conquer the whole world! Humble love is a frightening force, of which there is nothing stronger.



Here is a simpler version of the paragraph:

When you see something bad, like sin, you might be confused. Ask yourself if you should use force or love to fight it. Always choose love. If you do, you can win against anything. Love is the strongest power.

Figuur (2.5)

Afbeelding van Gooding 2022



Can you simplify the following paragraph in three different versions:

At some ideas you stand perplexed, especially at the sight of sin. And you must ask yourself whether to combat it with force or with humble love. Always decide to combat it with humble love. If you decide this once and for all, you can conquer the whole world! Humble love is a frightening force, of which there is nothing stronger.



Here are three simplified versions of the paragraph:

1. When you see something bad, like sin, you might be confused. Ask yourself if you should use force or love to fight it. Always choose love. If you do, you can win against anything. Love is the strongest power.
2. When you encounter ideas that confuse you, especially when it comes to sin, you may be unsure how to react. You must ask yourself if it is better to fight back with force or with humble love. It is always better to choose humble love. If you make this decision once and for all, you can overcome anything. Humble love is a powerful and intimidating force that is stronger than anything else.
3. If you see something bad, like sin, and you don't know what to do, remember that love is always the answer. Choose love over force. If you do, you will be able to conquer anything. Humble love is the most powerful force in the world.

Figuur (2.6)

Afbeelding van Gooding 2022



Which of the three simplified versions is the simplest?



Of the three simplified versions, the first one is the simplest. It uses the shortest and simplest words and phrases to convey the main idea of the original paragraph, which is that humble love is the strongest force and can be used to overcome anything. The other two versions add more details and examples, which make them slightly more complex than the first version. However, all three versions are simpler than the original paragraph and effectively convey the same idea.

Figuur (2.7)

Afbeelding van Gooding 2022.

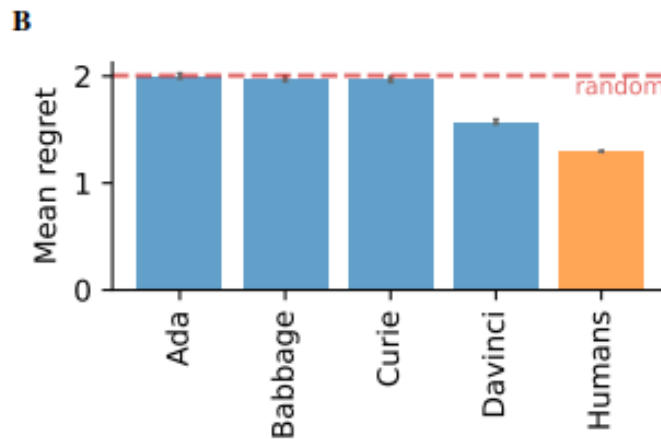
OpenAI biedt documentatie aan voor het GPT-3 taalmodel met vier verschillende engines, namelijk Davinci, Curie, Babbage en Ada. In maart 2023 is daar een vijfde engine bij gekomen, namelijk GPT-3 Turbo die wordt gebruikt door Chat-GPT. Davinci-003 is het meest geavanceerde model, geschikt voor taken zoals het schrijven van essays en het genereren van code, en levert de meest menselijke antwoorden. Curie is goed in nuance, maar minder menselijk dan Davinci, terwijl Ada en Babbage minder krachtig zijn en beter geschikt zijn voor eenvoudige taken zoals het aanvullen van tekst en sentimentanalyse (Greg e.a., 2023). Deze engines maken gebruik van dezelfde set hyperparameters, die kunnen worden aangepast en staan opgesomd in tabel 2.11.

Parameter	Omschrijving
model	Het GPT-3 model om te gebruiken: davinci, curie, babbage, ada, text-davinci-002, text-curie-001, text-babbage-001, text-ada-001, davinci-codex
temperature	De gulzigheid van een generatief model. Een lagere waarde zal conservatieve en voorspelbare tekst teruggeven. Hogere waarden zullen meer gevarieerde en onverwachtse tekst teruggeven, wat beter werkt bij creatieve toepassingen: een kommagetal tussen 0 en 1.
max_tokens	Het maximaal aantal tokens (woorden of subwoorden) dat het generatief model kan teruggeven: een getal tussen 1 and 2048.
top_p	Vergelijkbaar met temperature, maar deze waarde onderhoudt de probability distribution voor common tokens. Hoe lager de waarde, hoe waarschijnlijker de woordenschat dat het model zal overwegen bij het genereren van tekst. Een hoge waarde is toepasselijker wanneer een toepassing gericht is op nauwkeurigheid en correctheid: een kommagetal tussen 0 en 1.
stop	Een tekstwaarde (woord/symbool) tot waar het model zal genereren. When the model generates a string that matches any of the specified strings, it stops generating text: een lijst van string-waarden, of een enkele string.
presence_penalty	Factor die bepaalt hoe regelmatig woorden voorkomen: een kommagetal tussen 0 en 1

Tabel 2.11: Tabel met alle GPT-3 parameters.

Er zijn al diverse vergelijkende onderzoeken uitgevoerd naar de mogelijkheden van de OpenAI's ChatGPT en GPT-3 modellen, hoewel deze nog volop in ontwikkeling zijn. Een studie uitgevoerd door Binz en Schulz (2023) toonde aan dat de menselijkheid van de antwoorden die worden gegenereerd door de modellen onder de davinci-engine, werden beoordeeld op basis van de *mean-regret* score. Uit deze studie bleek dat deze modellen in staat zijn om redelijk menselijke antwoorden te produceren, zoals geïllustreerd in figuur 2.8. Uit het experiment van Tanya Goyal (2022) blijkt dat *zero-shot* samenvattingen met GPT-3 beter presteren dan *fine-tuned* modellen. Ook zijn er al enkele tools beschikbaar die gebruikmaken van de GPT-3 API, waaronder Jasper AI en ChatSonic, zoals aangehaald door Mottesi (2023). Voor het onderwijs zijn er mogelijkheden dankzij de hoge toegankelijkheid en granulaire personalisatie van het GPT-3 model, aldus Roose (2023) en Garg (2022). Wel is het zo dat GPT-3 niet geschikt is voor alle taken, zoals sentimentana-

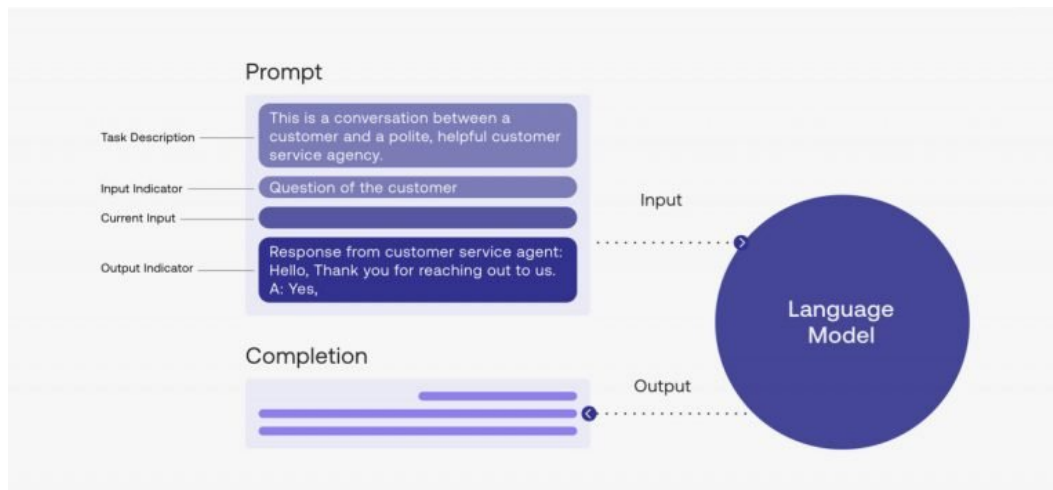
lyse en -classificatie, waarvoor een kleinschaliger taalmodel beter presteert (C. Li, 2022). Ten slotte wordt er ook aandacht besteed aan de ecologische effecten van de grote omvang van deze modellen, waarvoor alternatieve oplossingen zoals het gebruik van Cloud-infrastructuur en geschikte model finetuning worden voorgesteld, volgens Strubell e.a. (2019) en Simon (2021).



Figuur (2.8)

Experiment uit Binz en Schulz (2023) dat de mean-regret van de vier GPT-3 engines uittest.

De generatie van tekst en karakters door LLM's wordt gebaseerd op de waarschijnlijke uitkomst van een gegeven input. LLM's zoals GPT-3, BERT en T5 zijn bekend en dienen vaak als basis voor gefinetunde modellen. Door gebruik te maken van een neurale netwerk, kan het model patronen in de input herkennen en deze gebruiken om voorspellingen te doen over de output (Liu e.a., 2020). Het schrijven van een input of prompt is voor iedereen toegankelijk en intuïtieve tools zoals chatbots zijn ontworpen om een breed publiek te bereiken (McFarland, 2023). In dit kader is prompt engineering een belangrijke vaardigheid om effectief te communiceren met LLM's, zoals ChatGPT (Harwell, 2023). De vaardigheid van prompt engineering wordt geïllustreerd in figuur 2.9.

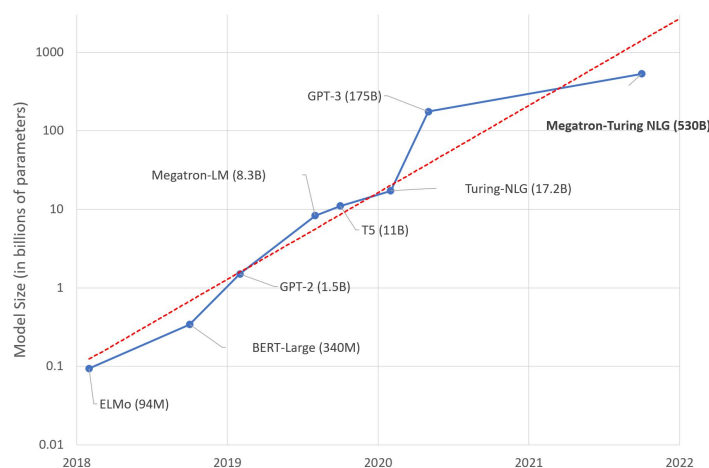
**Figuur (2.9)**

Afbeelding uit McFarland (2023)

Volgens onderzoek van Liu e.a. (2020) kunnen prompts worden gebruikt om werk te produceren dat is aangepast aan het doel. Het is belangrijk om een concrete en geoptimaliseerde prompt te hebben, inclusief een duidelijke scope, specifieke sleutelwoorden, de context en gepersonaliseerde keuzes (McFarland, 2023). Bij het opstellen van een prompt voor een zoekopdracht is het cruciaal om voldoende parameters op te nemen om te voorkomen dat het model te algemeen blijft en afwijkt van de intentie van de gebruiker. Effectieve prompt engineering voor AI leidt tot hoogwaardige trainingsgegevens, waardoor het AI-model nauwkeurige voorspellingen en beslissingen kan maken (Liu e.a., 2020). Prompt patterns zijn ontstaan samen met prompt engineering en lijken op software patterns. Deze patronen bieden herbruikbare oplossingen voor veelvoorkomende problemen in een bepaalde context, vooral bij de interactie met LLM's. White e.a. (2023) benoemt vier *prompt patterns*:

- Intent-prompts waarbij een LLM een instructie krijgt met een specifiek verwacht antwoord.
- Restriction-prompts die het antwoord van een LLM inperkt. Deze pattern is noodzakelijk om een LLM binnen de lijnen te houden.
- Contextualization-prompts verzekeren dat de output van een LLM relevant is. Een context wordt aan de LLM meegegeven.
- Expansion/reduction-prompts genereren een beknopte output met voldoende details.

BERT is een meertalige LLM die gebruik maakt van contextual word embeddings. Het model is getraind op 110 miljoen parameters uit 104 verschillende talen, waaronder Nederlands. Voor de Nederlandse taal zijn er twee varianten van BERT beschikbaar: RobBERT en BERTje. RobBERT wordt beschouwd als krachtiger dan BERTje. Het belangrijkste verschil tussen GPT-3 en BERT is volgens Mottesi (2023) de architectuur. GPT-3 is een autoregressief model dat alleen rekening houdt met de linkercontext bij het voorspellen of genereren van tekst, terwijl BERT bidirectioneel is en zowel de linker- als de rechtercontext in overweging neemt. De bidirectionele werking van BERT is geschikt voor sentimentanalyse, waarbij begrip van de volledige zincontext noodzakelijk is. Omdat GPT-3 toegang heeft tot meer informatie (45TB) dan BERT (3TB), kan dit een voordeel bieden tijdens taalbewerkingen zoals vereenvoudigen, parafraseren of vertalen. Ten slotte verschillen de LLM's ook qua grootte. Hoewel beide modellen erg groot zijn, is GPT-3 aanzienlijk groter dan BERT, mede door de uitgebreide trainingsdatasetgrootte (Brown e.a., 2020). Er is echter een nieuw generatief taalmodel genaamd LLaMa, dat sterker is dan GPT-3 en vergelijkbare modellen, terwijl het slechts tien keer minder parameters gebruikt. Helaas is LLaMa momenteel nog niet beschikbaar als online webtoepassing of API (Hern, 2023; Touvron e.a., 2023). De evolutie van pre-trained taalmodellen wordt in 2.10 weergegeven. De performantie van de modellen ten opzichte van de grootte volgt een lineaire functie.

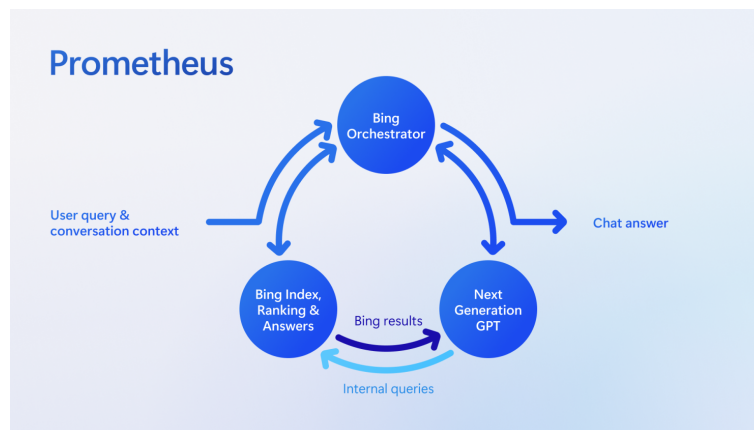


Figuur (2.10)

Afbeelding van Simon (2021)

Microsoft en OpenAI hebben een nauwe samenwerking waarbij het conversationele taalmodel van Bing gebruikmaakt van GPT-3. Door de Prometheus-technologie, visueel weergegeven in figuur 2.11, van Microsoft kan deze chatbot verder bouwen en verwijzingen bieden naar andere websites (Ribas, 2023). Deze verwijzingen zijn

mogelijk omdat Prometheus de Bing-index-, ranking- en antwoordresultaten combineert met het redeneervermogen van OpenAI's GPT-modellen. Via de *Bing Orchestrator* genereert Prometheus iteratief een set interne queries om zo binnen de gegeven gesprekscontext nauwkeurige antwoorden op gebruikersvragen te bieden (Ribas, 2023). Bing AI is momenteel in de testfase en is beschikbaar als webpagina en browserextensie voor Microsoft Edge. Het gebruik van extraherende en abstraherende samenvattingen maakt de chatbot interessant, maar er is nog onderzoek nodig naar de credibiliteit en correctheid van de verstrekte verwijzingen. Het is belangrijk om op te merken dat er geen officiële API beschikbaar is en de limiet tijdens een conversatie beperkt is tot 2000 tokens, in tegenstelling tot GPT-3.



Figuur (2.11)

Afbeelding van Ribas (2023)

Experten halen het GPT-3 model en ChatGPT aan als de toekomst voor gepersonaliseerde en adaptieve uitleg aan scholieren. Bing AI biedt een extra dat revolutionair kan zijn bij het opzoeken van uitleg voor zoektermen, zonder het verlies aan bronvermelding. Huidige toepassingen staan mogelijks in een spreekwoordelijke schaduw eenmaal leessoftware voor scholieren met dyslexie worden ontwikkeld met AI. De mogelijkheden van GPT-3 zijn eindeloos en toepassingen die hiervan gebruik maken, kunnen in het onderwijs ingezet worden als ondersteunende software.

2.6. De valkuilen bij AI en NLP.

AI en ML zijn volop in groei. NLP gebruikt AI en ML om menselijke taal te verwerken, terwijl NLU deze technologieën gebruikt om menselijke taal te begrijpen. Hoewel deze technologieën veelbelovend zijn, moeten AI-ontwikkelaars rekening houden met veelvoorkomende en genegeerde uitdagingen en valkuilen (Khurana e.a., 2022; Roldós, 2020; Sciforce, 2020). Deze sectie beantwoordt de volgende onder-

zoeksvraag:

- Met welke valkuilen bij taalverwerking met AI moeten ontwikkelaars rekening houden?

Het ontwikkelen van NLP- en NLU-toepassingen is duur en vormt een obstakel voor veel IT-professionals vanwege factoren zoals het gebrek aan NLP-expertise, de kwaliteit en kwantiteit van data, de integratie en deployment van modellen en de transparantie van modellen (IBM, 2022). Bij de ontwikkeling en finetuning van een NLP-toepassing met AI verkiezen software-ontwikkelaars *black-box* modellen. Hoewel het verschil in nauwkeurigheid minimaal is, wordt de afweging gemaakt op basis van de transparantie van het model. Na een transformatie wordt niet aangegeven waarom specifieke transformaties zijn uitgevoerd, zoals het vervangen van een woord door een eenvoudiger synoniem. *White-box* taalmodellen zijn schaars (Sikka & Mago, 2020).

Sequence labeling kan worden bemoeilijkt door homoniemen, zoals beschreven in onderzoek door Roldós (2020). Een voorbeeld hiervan is het woord 'bank', dat voor de machine niet duidelijk aangeeft of het gaat om de financiële instelling of het meubelstuk. Methoden zoals Word Sense Disambiguation (WSD), PoS-tagging en contextual embeddings kunnen de betekenis van woorden bepalen op basis van de context (Eisenstein, 2019; Liu e.a., 2020). Het verbeteren van NLP-systemen met synoniemen en antoniemen kan worden bereikt door het gebruik van candidate generation en synonym detection, terwijl meertalige transformers zoals BERT een oplossing bieden voor de beperkte toepassingen in andere talen dan het Engels (Dandekar, 2016; Roldós, 2020).

Verschillende studies hebben aangetoond dat tekstvereenvoudiging bedoeld is om gelijke kansen te bieden aan iedereen. Bij het ontwikkelen van adaptieve tekstvereenvoudigingstoepassingen is het echter belangrijk om ethische overwegingen en bewustzijn van de behoeften van de eindgebruiker in acht te nemen, zoals beschreven in onderzoeken van Niemeijer e.a. (2010), Xu e.a. (2015), en Gooding (2022). Om de eindgebruiker meer controle te geven, moet deze de keuze hebben om te kiezen welke delen van de tekst vereenvoudigd moeten worden, wat kan worden bereikt door synoniemen te gebruiken of zinnen te markeren die moeilijk te begrijpen zijn.

Hoewel iedereen kan converseren met een chatbot, vereist het verkrijgen van gepaste en verwachte antwoorden een doordachte input. Onnauwkeurige prompts of een gebrek aan trainingsdata kunnen leiden tot onjuiste output. Door het gebruik van conditionele expressies of het finetunen van hyperparameters kan echter

de betrouwbaarheid van de antwoorden worden vergroot (Jiang, 2023; Miszczak, 2023).

Het beoordelen van vereenvoudigde teksten vereist de nodige opvolging van ontwikkelaars in vergelijking met andere ML- of NLP-taken. Evaluatiemetrieken zoals ROUGE en BLEU zijn beperkt omdat ze geen rekening houden met de semantiek tussen een referentietekst en een vereenvoudigde of samengevatte tekst. Om dit probleem op te lossen, beveelt Fabbri e.a. (2020) aan dat ontwikkelaars menselijke evaluatie inschakelen om de vereenvoudigde tekst van een taalmodel te beoordelen. De onderzoekers dringen aan op verdere studie naar nieuwe standaarden en beste praktijken voor betrouwbare menselijke beoordeling. Bovendien moeten de doelgroepen voor wie de tekst wordt vereenvoudigd, nauw betrokken worden bij het proces (Iskender e.a., 2021). Tabel 2.12 geeft een synthese schema met prevalente struikelblokken bij de ontwikkeling van NLP-toepassingen.

Probleem	Oplossing
Dure ontwikkeling en onderhoud van taalmodellen	Voorkeur voor black-box modellen bij ontwikkeling en finetuning. API's kunnen als alternatief dienen op zelf-gehoste taalmodellen.
Homoniemen kunnen sequence labeling bemoeilijken	Word Sense Disambiguation, PoS-tagging en contextual embeddings.
Paternalisme	Ontwikkeling van gepersonaliseerde tekstvereenvoudigingstoepassingen moeten de eindgebruiker meer controle geven, zoals het kiezen welke delen van de tekst vereenvoudigd moeten worden, het gebruik van synoniemen of het markeren van zinnen die moeilijk te begrijpen zijn.
Onnauwkeurige prompts	Gebruik van conditionele expressies bij prompts of one-shot summary uitvoeren.
Onnauwkeurige evaluatie van tekstvereenvoudiging	Menselijke evaluatie toepassen of gebruik maken van ROUGE-L metrieken die wel de semantiek in acht nemen.

Tabel 2.12: Samenvattend schema met vaak voorkomende struikelblokken bij NLP-toepassingen.

2.7. Conclusie

Wetenschappelijke artikelen intensief lezen is voor scholieren met dyslexie in de derde graad van het middelbaar niet enkel moeizaam, maar gaat verder dan dat. Deze scholieren kunnen moeite ondervinden bij het ontcijferen en automatiseren van woordherkenning. MTS en aangepaste visuele weergaven bieden bewezen voordelen voor scholieren met dyslexie, maar de leesbaarheid van wetenschappelijke artikelen neemt af. Het gebruik van vakjargon, ingewikkelde woordenschat en moeizame syntax sluiten een algemeen publiek uit en maken het enkel mogelijk voor wetenschappelijk geletterden om deze artikelen te lezen. Een uniform formaat biedt echter kansen voor geautomatiseerde vereenvoudiging van tekst, waarbij experts tactieken hebben ontwikkeld om teksten te vereenvoudigen op maat voor scholieren met dyslexie, zoals het toepassen van leesbaarheidsformules of intuïtie, en het vereenvoudigen van zinnen op lexicaal, syntactisch en semantisch niveau en samenvattingen. Er zijn taalmodellen beschikbaar in de vorm van API's of open-source software die deze transformaties kunnen uitvoeren. Momenteel fungeert software die door de overheid wordt uitgeleend aan scholieren met dyslexie voornamelijk als voorleessoftware, maar nieuwe technologieën zoals GPT-3 en LLM's bieden mogelijkheden voor tekstvereenvoudiging. Ontwikkelaars moeten zich echter bewust zijn van het feit dat andere taalmodellen zoals BERT voor taken zoals semantische analyse soms betere resultaten leveren met minder rekenkracht.

3

Methodologie

Hiervoor moeten drie onderzoeksmethoden toegepast worden, met als hoofddoel het ontwikkelen van een prototype voor gepersonaliseerde ATS om wetenschappelijke artikelen te vereenvoudigen voor scholieren met dyslexie in de derde graad van het middelbaar onderwijs.

3.1. Requirementsanalyse

Om *must-haves* van *wont-haves* bij de ontwikkeling van een prototype voor tekstvereenvoudiging te kunnen onderscheiden, moet het onderzoek stilstaan bij de vereiste functionaliteiten door middel van een requirementsanalyse. Ten eerste moet deze onderzoeksfase zich richten op het uittesten van een shortlist van tools met tekstvereenvoudigingsfunctionaliteiten op hun capabilities voor gepersonaliseerde ATS. Bovendien moeten zowel erkende toepassingen van de overheid gericht op deze casus, alsook toepassingen die leerkrachten of scholieren kunnen gebruiken om teksten te vereenvoudigen, geanalyseerd worden. Deze analyse verduidelijkt hoe sterk deze toepassingen aanleunen op een tekst vereenvoudigd door een leerkracht die een gepersonaliseerde tekstvereenvoudiging maakt van een wetenschappelijk artikel, specifiek op maat voor een scholier met dyslexie. Op deze manier kan de requirementsanalyse een antwoord geven op de volgende twee deelvragen van het onderzoek, aan de hand van de technieken voor MTS en ATS beschreven in hoofdstuk 2.

- Welke functies ontbreken AI-toepassingen om geautomatiseerde tekstvereenvoudiging mogelijk te maken voor scholieren met dyslexie in de derde graad middelbaar onderwijs?
- Welke manuele methoden voor tekstvereenvoudiging komen niet in deze tools voor?

Zoals aangewezen in sectie 2.5 beschikken scholen over vijf door de overheid erkende softwarepakketten, maar de requirementsanalyse neemt enkel drie van deze vijf op omwille van hun prevalentie toepassing in het onderwijs. Daarnaast kunnen online beschikbare tools ook scholieren met dyslexie ondersteunen bij het intensief lezen van complexe teksten met ATS, zoals bewezen in Bingel e.a. (2018). De requirementsanalyse betreft enkel tools met onderschreven tekstvereenvoudigingscapaciteiten en laat zo pure samenvattingstools erbuiten. Deze shortlist van tools worden weergegeven in 3.1.

Erkende software	Online beschikbare tools
Sprintplus (ES1)	Simplish (OB1)
Kurzweil3000 (ES2)	SciSpace (OB2)
AlineaSuite (ES3)	ChatGPT (OB3)
	Bing Chatbot (OB4)

Tabel 3.1: Shortlist van uit te testen tools en toepassingen voor tekstvereenvoudiging.

De opgesomde MTS-technieken uit 2.5 en ?? dienen als bouwsteen om de shortlist te valideren op bruikbaarheid voor gepersonaliseerde ATS binnen het onderwijs, specifiek voor scholieren met dyslexie in de derde graad van het middelbaar onderwijs. Met behulp van de criteria beschreven in tabel ?? past de requirementsanalyse deze technieken toe om de gebruikte en ontbrekende MTS-technieken in de tools te achterhalen, alsook of deze tools in staat zijn om wetenschappelijke artikelen te kunnen vereenvoudigen.

Functionaliteiten worden teruggevonden op basis van een verkenning in de tool en alle aangeboden functies komen aan bod in deze onderzoeksfase. Artikelen van ... en ... dienen als testmiddel voor deze onderzoeksfase. De tools ondergaan een evaluatie om hun vermogen om een wetenschappelijk artikel in PDF-formaat te vereenvoudigen, indien dat mogelijk is. Als dit niet mogelijk is, gebeurt het kopiëren en plakken van de tekst in de toepassing. Vervolgens vindt een evaluatie van de eventuele beperkingen of tekortkomingen plaats.

Vorm van MTS	Functionaliteit
Lexicale vereenvoudiging	<p>Rekening houden met doelgroep buiten vakgebied door eenvoudigere synoniemen te schrijven</p> <p>Woorden met minder lettergrepen gebruiken</p> <p>Extra uitleg schrijven bij zinnen</p> <p>Paragrafen herschrijven zodat ze eerst uitleg geven op een high-level niveau, vervolgens lagen van complexiteit toevoegen om de lezer te begeleiden</p> <p>Woordenlijst aanmaken</p> <p>Synoniemenlijst aanmaken</p> <p>Idiomen vervangen door eenvoudigere synoniemen</p>
Syntactische vereenvoudiging	<p>Zinnen inkorten</p> <p>Verwijswoorden aanpassen</p> <p>Voorzetseluitdrukkingen</p> <p>Samengestelde werkwoorden aanpassen</p> <p>Actieve stem toepassen</p> <p>Onregelmatige werkwoorden gebruiken</p>
Formaatwijzigingen	<p>Achtergrondkleur aanpassen</p> <p>Woord- en karakterspatiering</p> <p>Consistente lay-out</p> <p>Duidelijk zichtbare koppenstructuur</p> <p>Huidige positie benadrukken</p> <p>Waarschuwingen geven omtrent formulieren en sessies</p> <p>Inhoud visueel groeperen</p> <p>Tekst herschrijven als tabel</p> <p>Tekst herschrijven als opsomming</p>

Tabel 3.2: Criteria waarop toepassingen worden afgetoetst in de requirementsanalyse

MoSCoW-principe	Functionaliteit
Must-have	<p>Gepersonaliseerde vereenvoudiging aanbieden, waaronder lexicale en syntactische vereenvoudiging aanbieden, na het toevoegen van een respectievelijke API-sleutel.</p> <p>Wetenschappelijke artikelen in PDF-vorm opladen.</p> <p>Personaliseerbare site: lettertype -en grootte aanpassen, tekst-formaat aanpassen, achtergrondkleur aanpassen</p> <p>Lokale opzet</p>
Should-have	<p>Glossary genereren na handmatige selectie van moeilijke woorden</p> <p>Personaliseerbare PDF- of Worddocumentlay-out</p> <p>Uitvoer als PDF of Word-bestand teruggeven.</p> <p>Tekstanalyse voor en na de vereenvoudiging aanbieden.</p>
Could-have	Glossary genereren na automatische selectie van moeilijke woorden
Wont-have	<p>Beschikbaarheid tot de tool zonder Docker Desktop, in de vorm van online webtoepassing of browserextensie.</p> <p>Beschikbaarheid tot de standaard- en gepersonaliseerde opties zonder API-sleutels</p>

Tabel 3.3: Het Moscow-schema, opgebouwd door middel van de requirementsanalyse.

3.2. Vergelijkende studie

Teksten vereenvoudigen met ATS kan niet efficiënt verlopen zonder een taalmodel, maar een toepassing voor tekstvereenvoudiging binnen de casus van wetenschappelijke artikelen moet gebruik maken van een taalmodel die het meest aansluit op deze casus. Om het prototype af te stemmen op één taalmodel, vereist er een antwoord op de volgende vraag.

- Welk taalmodel of LLM is geschikt voor ATS van wetenschappelijke artikelen voor scholieren met dyslexie in de derde graad van het middelbaar onderwijs, met dezelfde of gelijkaardige kwaliteiten als gepersonaliseerde tekstvereenvoudiging met MTS?

Gezien de schaarse hoeveelheid aan gespecialiseerde taalmodellen, specifiek gericht op het vereenvoudigen van wetenschappelijke artikelen of taalmodellen met een personaliseerbaar karakter, beoordeelt de vergelijkende studie alle vermelde taalmodellen die specifiek op ATS gericht zijn, beschreven in 2.10.

Verwijzing	Taalmodel
T1	Haining Scientific Abstract Simplification ¹
T2	Keep It Simple ²
T3	GPT-3

Tabel 3.4: Gebruikte taalmodellen in de vergelijkende studie

Zo vergelijkt deze onderzoeksfase de leesgraadsmetrieken van de oorspronkelijke wetenschappelijke artikelen zoals in 3.1, met referentietekst vereenvoudigd door MTS en teksten vereenvoudigd met ATS.

De referentieteksten zijn geschreven door twee leerkrachten binnen het vakgebied onderwijs, en twee scholieren van de derde graad in het middelbaar onderwijs zonder dyslexie. Deze vier personen baseren zich op vooraf meegekregen richtlijnen, toegelicht in de bijlage.

Zinnen worden vereenvoudigd om zo te vergelijken welk taalmodel beter in staat is voor gepersonaliseerde tekstvereenvoudiging. Hiervoor zijn vier scripts nodig.

Eerst volgt er een voorbereiding van de tekstinhoud. Hier wordt namelijk de inhoud van de wetenschappelijke artikelen opgehaald, en vervolgens in een tekstbestand geplaatst. Dit gebeurt manueel. Dit gebeurt met de code verwezen in

B.1. Aangezien GPT-3 en de HuggingFace-taalmodellen enkel per commandline of Python-script werken, krijgt het taalmodel alle bruikbare testinhoud in *plain-text*. Deze bruikbare tekstinhoud omvat alle tekst uit het wetenschappelijk artikel, met uitzondering op de bibliografie en inhoudsopgave.

Zoals eerder verwezen, moeten de taalmodellen via HuggingFace eerst vertaald worden naar Engelstalige tekst. Zo staat het vertalen in de tweede fase centraal, verwezen in script B.2. Aangezien het onderzoek probeert te werken met zo vaak mogelijk open-source of vrij beschikbare software, wordt er gebruik gemaakt van de *deep_translator* Python-bibliotheek. Het script itereert iedere zin en vertaalt de zin naar het Engels, indien dit een niet-Engelstalige zin is. Het resultaat van dit script is een csv-bestand met daarin twee kolommen, alle Nederlandstalige en alle vertaalde Engelstalige zinnen van één artikel. Bij een Engelstalig artikel bevatten de twee kolommen dezelfde waarden. De kolommen worden gesepareerd met een pipe-symbool.

Vervolgens wordt iedere zin aan het taalmodel gegeven, zo om voor elke zin een vereenvoudigde versie te krijgen. T1 en T2 vereenvoudigen de Engelstalige zinnen, in tegenstelling tot T3 en de verwante prompts die de Nederlandstalige zin vereenvoudigen. Bij het schrijven van de prompt wordt er rekening gehouden met de verwachte *output format*.

Eerst worden de specifieke tokenizers en modellen van T1 en T2 gedeclareerd, alsook wordt de API-sleutel voor GPT-3 ingesteld. Het script itereert over alle tekstbestanden aangemaakt in fase 2 of in script B.2.

Taalmodellen via HuggingFace worden lokaal opgezet en worden niet aangepast. Tabel ?? geeft de gebruikte prompts voor de testen bij het GPT-3 model weer. Zowel het Seq2Seq als het toegepaste GPT-3 model maken gebruik van een nul-temperature en een top-p waarde van 90%, zo om een probabilistisch vertrouwd antwoord te verkrijgen, alsook om een hoge woordfrequentie te bekomen, zoals eerder aangegeven in 2.11.

Zoals eerder aangehaald kunnen prompt-gebaseerde testen verschillende resultaten bekomen, afhankelijk van de gegeven input. Daarom benadert de vergelijkende studie vijf verschillende prompts, gebaseerd op de verschillende tekstvereenvoudigingstechnieken beschreven in 2.5. De tokenlengte kan een request doen falen. Daarom breekt het script de tekst per paragraaf op.

Naam	Prompt
P1	Vereenvoudig deze tekst
P2	Vereenvoudig deze tekst voor studenten (16-18 jaar) door moeilijke woorden te vervangen, vakjargon te schrappen, woorden langer dan 18 letters te vervangen, acroniemen voluit te schrijven, een woord slechts eenmaal door een synoniem te vervangen, korte uitleg te geven wanneer dat nodig is, en percentages te vervangen.
P3	Vereenvoudig een tekst door deze op te delen in kortere zinnen van maximaal tien woorden. Verander voornaamwoorden als 'zij', 'hun' of 'hij' in namen. Vervang complexe zinsconstructies en voorzetselzinnen door eenvoudiger alternatieven, maar laat ze ongewijzigd als er geen eenvoudiger optie beschikbaar is.

Tabel 3.5: De GPT-3-prompts die in de vergelijkende studie aan bod komen.

Leesgraadsformules dienen, zoals aangegeven in Nenkova en Passonneau (2004), als objectieve maatstaf bij deze vergelijkende studie. De bekomen leesgraadmetrieken ondergaan een vergelijking met elkaar en met die van een referentietekst. De vergelijking gebeurt zowel subjectief als objectief aan de hand van de leesgraadsmetrieken benoemd in tabel ???. Code-blok ??? past de Pandas-bibliotheek in combinatie met de Readability-library toe om een CSV te maken met alle leescriteria.

Naast leesgraadsmetrieken moeten de vereenvoudigde teksten eveneens de semantiek van de tekst kunnen behouden, alsook de volledige kern van de tekstinhoud nog steeds bijhouden. Zo moeten de referentieteksten capabel zijn om alle meegegeven vragen te beantwoorden.

De vergelijkende studie bepaalt welk taalmodel nodig is om gepersonaliseerde tekstvereenvoudiging in het prototype aan te bieden, nadrukkelijk om wetenschappelijke artikelen met ATS te vereenvoudigen op maat voor scholieren met dyslexie in de derde graad van het middelbaar onderwijs en met een vergelijkbare kwaliteit als een vereenvoudiging met MTS.

3.3. Prototype voor tekstvereenvoudiging

Met de benodigde functionaliteiten en de geschikte taalmodel voor gepersonaliseerde ATS, kan het onderzoek een volgende stap zetten richting de onderzoeksvraag. Deze sectie omschrijft de ontwikkeling van een prototype voor gepersonaliseerde tekstvereenvoudiging voor scholieren met dyslexie in de derde graad van

het middelbaar onderwijs en geeft daarmee een antwoord op de volgende deelvraag:

- Hoe kan een intuïtieve en lokale webtoepassing worden ontwikkeld die zowel scholieren met dyslexie als docenten helpt bij het vereenvoudigen van wetenschappelijke artikelen met behoud van semantiek, jargon en zinsstructuren?

Allereerst moeten de gebruikte technologieën benadrukt worden. Eerdere onderzoeken maakten sporadisch gebruik van Python-technologieën en dit prototype volgt deze voorafgaande onderzoeken en vakliteratuur. Tabel ?? geeft een breed overzicht van alle gebruikte programmeertalen. Hierop vult tabel ?? aan door een overzicht te geven van alle gebruikte Python-libraries.

Technologie	Functionaliteit
Python	De back-end van het prototype die API-calls en de NLP-functionaliteiten zoals PoS-tagging en lemmatizing verwerkt.
JavaScript	De toepassing gebruikersvriendelijker maken, personalisatie-opties voor de site doorvoeren en de functies gebouwd in Javascript dienen als alternatief op commandline instructies.
HTML en CSS	Het visuele uiterlijk van de website aanpassen naargelang de gekozen parameters van de eindgebruiker.
Jinja	Informatie uit de back-end (Python) doorgeven aan de front-end (JavaScript).
Docker	Lokale uitrol van de webtoepassing.
Bash	Intuïtief script om de webtoepassing op te starten voor Linux en Mac-systemen.
Powershell	Intuïtief script om de webtoepassing op te starten op Windows-systemen.

Tabel 3.6: Gebruikte programmeertalen in het prototype voor tekstvereenvoudiging.

Python-bibliotheek	Functionaliteit
Flask	Het framework van de webtoepassing. Deze combineert front-end en back-end en past binnen de scope van een prototype.
PDFMiner	Tekstinhoud van PDF's inlezen.
EasyOCR	PDF-pagina's inscannen als afbeelding in JPG-formaat om vervolgens de tekst te extraheren.
NumPy	De reshape-functie vereenvoudigt de manier om arrays van zinnen bij elkaar te plaatsen om zo een paragraaf te bekomen.
Spacy	PoS-tagging en het lemmatiseren van woorden.
OpenAI	GPT-3 API aanspreken.
Requests	HuggingFace API aanspreken en informatie ophalen van een lexicale databank API.
BeautifulSoup	HTML-content <i>parsen</i> zodat de aangepaste inhoud van een leerkracht correct geformatteerd kan worden naar een PDF.

Tabel 3.7: Gebruikte Python-libraries en hun respectievelijke functie in het prototype.

De homepage, weergegeven op figuur 3.1, biedt een kort overzicht van de twee algemene functionaliteiten, alsook instellingen waar gebruikers de webtoepassing kunnen aanpassen naargelang hun voorkeuren. Zoals aangeraden in 2 door 2023, moeten ontwikkelaars rekening houden met de personalisering van de website. Deze functionaliteit gaat volgens hem vaak onopgemerkt en heeft nochtans een bewezen effect op het leesgedrag -en begrip van zowel mensen met als zonder dyslexie. De webtoepassing maakt gebruik van de standaardparameters, uitgewezen in Rello en Baeza-Yates (2013) en Rello, Baeza-Yates, Dempere-Marco e.a. (2013). Op een apart scherm kan de eindgebruiker de volgende elementen aanpassen naar hun toebehoren. Het prototype kan na aanpassing van de parameters eruit zien zoals weergegeven in figuur 3.2. JavaScript maakt het mogelijk om deze parameters dynamisch en on-the-spot aan te passen. Na een aanpassing zal de back-end de sessievariabelen aanpassen naargelang de gekozen parameters, zodat de eindgebruiker niet per pagina deze parameters moet instellen. Voor dit prototype zijn er twee soorten eindgebruikers: leerkrachten die wetenschappelijke artikelen met ATS willen vereenvoudigen op maat van scholieren en de scholieren die dit zelf willen doen. Beide doelgroepen hebben hun eigen noden, maar toch een centraal doel voor ogen: het gepersonaliseerd vereenvoudigen van een wetenschappelijk artikel.

Prototype voor tekstvereenvoudiging

Voor leerlingen	Voor lectoren
Voeg een PDF toe: <input type="button" value="Bestand kiezen"/> <small>Geen bestand geselecteerd</small> Geavanceerde upload <input type="checkbox"/> <input type="button" value="View PDF"/>	Voeg een PDF toe: <input type="button" value="Bestand kiezen"/> <small>Geen bestand geselecteerd</small> Geavanceerde upload <input type="checkbox"/> <input type="button" value="View PDF"/>
Voer tekst in: <input type="checkbox"/> <input type="button" value="View PDF"/>	Voer tekst in: <input type="checkbox"/> <input type="button" value="View PDF"/>

Uitleg prototype

Voor scholieren biedt een online webpagina aan waarop een scholier zinnen of paragrafen kan vereenvoudigen naar keuze.

Voor lectoren biedt u de mogelijkheid aan om een nieuw PDF-bestand aan te maken vanuit het oorspronkelijke PDF-bestand.

Bij **instellingen** kan u de weergegeven tekst binnen de online tool aanpassen volgens uw wens.

Instellingen

Figuur (3.1)

Een mogelijke weergaven van de homepagina.

The settings page is divided into three main sections:

- Achtergrondkleur**: A color selection tool with a small square preview.
- Tekstkenmerken**: A collection of controls for text formatting:
 - Lettertype**: A dropdown menu currently set to 'Arial'.
 - Woord Spatiering**: A slider ranging from 0 to 0.5.
 - Regelhoogte**: A slider ranging from 0 to 1.
 - Lettergrootte**: A slider ranging from 0 to 21.
 - Aligning**: A dropdown menu currently set to 'Links'.
 - A button labeled 'Instellingen aanpassen'.
- API-sleutels**: Two input fields labeled 'OpenAI API KEY' and 'HuggingFace API KEY'.

Figuur (3.2)

Voorbeeldweergave van de instellingenpagina.

De invoer van een wetenschappelijk artikel gebeurt voor beide doelgroepen op identieke wijze. De gebruiker kan een PDF of een stuk *plain-text* ingeven als invoer van het wetenschappelijk artikel. Eindgebruikers kunnen met het prototype

wetenschappelijke artikelen op één van twee manieren inladen: *plaintext* of via een PDF-bestand. PDF's worden tijdelijk *in-memory* opgeslaan. Daarmee kan het systeem niet na lange duur overbelast raken. Het prototype past eenzelfde functie toe om wetenschappelijke artikelen in te lezen voor de twee verschillende doelgroepen. Vooraf controleert de Flask-applicatie het type invoer. Vervolgens spreekt de Flask back-end de Reader-klasse aan die de tekst verder zal formatteren tot een bruikbaar formaat voor de webtoepassing. PDF extractors, waaronder PDFMiner, kunnen tekstinhoud verliezen tijdens het extraheren zoals eerder aangewezen in tabel 3.1. Als vangnet biedt het prototype een tweede optie aan waarbij de PDF-pagina's als afbeelding worden opgeslaan. Er wordt rekening gehouden met de splitsing tussen normale en geavanceerde upload. Deze twee methoden zijn terug te vinden in de Reader-klasse³ ofwel bijlage C.3.

1. PDFMiner itereert doorheen de PDF en extraheert vervolgens de tekst op iedere pagina. Deze methode resulteert in een string-object.
2. De Python-bibliotheek EasyOCR voorziet een eenduidige en ontwikkelaarsvriendelijke manier om PDF-pagina's op te slaan als JPG of PNG. Tesseract biedt een even eenduidige oplossing aan, maar vergt meer configuraties en daarmee vergroot de omvang van de Docker-container. Vervolgens worden de afbeeldingen per text-chunk ingelezen en nadien opgeslaan. Het gebruik van deze alternatieve methode kan de omvang van de Docker-container vergroten. Om dit te voorkomen, verwijdert het prototype afbeeldingen nadat deze de tekstinhoud hebben geëxtraheerd en opgeslaan om zo ruimte te besparen. Net zoals bij de eerste methode resulteert deze methode in een string-object.

De eerste fase van het prototype slaat de tekstinhoud op in meerdere arrays die zinnen voorstellen. Zoals aangegeven in C.3, staat de standaardparameter voor het aantal zinnen per paragraaf ingesteld op vijf zinnen, maar kunnen gebruikers aanpassen via het HTML-formulier bij de instellingen. Om de PoS-tag bij het respectievelijke woord bij te houden, maakt het prototype gebruik van een *key-value pair* datastructuur. Zo verwijst de sleutel naar het woord in een zin; de waarde verwijst naar de PoS-tag die aan het woord toebehoort. Het prototype is enkel ontworpen voor Nederlandstalige en Engelstalige wetenschappelijke artikelen. Daarmee laadt het enkel twee embeddingsmodellen van Spacy op, zoals vermeldt in tabel ???. Hardcoderen is uit den boze en daarom maakt het prototype gebruik van een *dictionary* die de naam van deze embeddingsmodellen bijhoudt, zoals aangegeven in C.3. Zo is het systeem en hoeft er enkel een taalherkenning plaats te vinden. Fouttolerantie aanbieden gebeurt in de vorm van een standaardtaal in de dictionary, namelijk het Nederlands, of door vooraf de gebruiker te vragen in welke taal de opgelade tekst staat via een HTML-formulier.

³<https://github.com/Dyashen/text-simplification-tool/blob/main/web-app/Reader.py>

Taal	Embeddingsmodel
Nederlands	NL Core News Medium ⁴
Engels	EN Core Web Medium ⁵

Tabel 3.8: Gebruikte SpaCy Word-embeddings

3.3.1. Tool voor leerkrachten

In de vorm van een HTML-formulier kunnen leerkrachten gepersonaliseerde ATS met dit prototype waarmaken. Op de overzichtspagina kunnen leerkrachten beschikken over de functionaliteiten, weergegeven in tabel ???. Het HTML-formulier omvat alle benodigde tekstvereenvoudigingstechnieken op lexicaal en syntactisch niveau die ??? verder uitwees.

Om afwijkende resultaten op een GPT-prompt te vermijden, wordt de temperature op nul geplaatst en de *top_p* waarde wordt ingeschat op 80%. SpaCy wordt gebruikt om woordkenmerken zoals de PoS-tag op te halen, maar het systeem is vatbaar voor het niet kunnen vinden van afwisselende en meertalige woordenschat. Een mogelijke oplossing is om de taal te veranderen naar Engels of Frans, of een aangepast taalherkenningsmodel te gebruiken. Een andere optie is om de tekst voor te verwerken om de Nederlandse en Engelse woorden te scheiden voordat ze worden verwerkt met SpaCy. Adjectieven uit de tekst verwijderen is mogelijk zonder taalmodel. Aangezien alle woorden gekoppeld worden aan een PoS-tag, is het eenvoudig om de woorden gelinkt aan de span-tag van de adjectieven uit te filteren.

De eenduidige HTML-structuur van online woordenboeken maken het mogelijk om gratis en eenvoudig de definities van woorden op te halen. Zo is het mogelijk om annotaties op te halen zoals aangewezen in het onderzoek van Bulté e.a. (2018). Met behulp van Requests en BeautifulSoup is het mogelijk om lijsten met definities te scrapen van deze sites. De stam van het gemarkeerde woord wordt opgehaald en vervolgens meegegeven als zoekopdracht. De bron wordt samen met het resultaat aan de eindgebruiker getoond.

Tekstvereenvoudiging

Het prototype gebruikt een taalmodel van *HuggingFace* voor extraherende samenvattingen en zowel gratis taalmodellen van *HuggingFace* als het GPT-3 taalmodel voor abstraherende samenvattingen. Het model kan parameters, zoals maximale lengte van de gegenereerde tekst, ontvangen en biedt zowel gepersonaliseerde als niet-gepersonaliseerde vereenvoudiging. Het gebruik van *HuggingFace* vereist

Functionaliteit	Gebruikte JS/Python-techniek
Specifieke meegeven per paragraaf	Naast een optie om voor het hele document één prompt te gebruiken, voegt het prototype ook een optie toe om. Hiervoor past de webinhoud een key-value structuur toe.
Opties voor gepersonaliseerde ATS aangeven.	Met behulp van een HTML-formulier kunnen leerkrachten checkboxes afvinken waaraan de gegenereerde tekst moet voldoen. Deze verschillende opties zijn terug te vinden in ?? .
Werkwoorden, bijvoeglijke en zelfstandige naamwoorden markeren	Front-end aanpassing met eventlistener. De tekstkleur van het aangegevoerde type woorden verandert naar het gekozen kleur.
Zinnen verwijderen	Front-end filter aangesproken door een <i>eventlistener</i> .
Woord toevoegen aan woordenlijst	Een eventlistener handelt de functionaliteit af door de woorden en hun zin van voorkomen tijdelijk op te slaan in een storage. Het formulier houdt dit bij en geeft het vervolgens mee bij het indienen. Deze woorden en hun respectievelijke zin van voorkomen dienen om de woordenlijst op te vullen in het gegenereerde PDF of Word-bestand.
API-sleutel en sessieherkenning	Detectie met JS of een sessievariabele bestaat.

Tabel 3.9: Alle beschikbare functionaliteiten in

een internetverbinding en kan geen extra trainingsdata bevatten. De opstarttijd voor alle *HuggingFace*-taalmodellen wordt bij de start van de applicatie afgehandeld door middel van een extra parameter de request. Sleutels worden standaard bijgehouden in env-bestanden. Via de webtoepassing kan een gebruiker deze sleutel aanpassen. Binnen een lokale omgeving is dit in orde, al moeten ontwikkelaars rekening houden met beveiligingsmaatregelen wanneer een dergelijke tool wordt uitgerold. Het merendeel van de gebruikte taalmodellen is Engelstalig of is nadrukkelijk getraind op basis van Engelstalige datasets. De ingegeven tekst wordt eerst vertaald naar het Engels om zo de kans op een accurate vereenvoudiging te verhogen. Voor de vertaling wordt de Google Translate Python-package gebruikt. Deze is minder accuraat vergeleken met DeepL, maar biedt een gratis beschikbaar en aanvaardbaar alternatief aan. Factoren zoals topic diversity en semantische redundantie moeten overwogen worden bij het kiezen van een taalmodel voor extraherend samenvatten. Lange documenten samenvatten kan zoals aangeduid in literatuurstudie door extraherende samenvatting, gevolgd door abstraherende samenvatting om de tekst coherent te doen blijken. Eerder werd er gekozen om de voltekst per paragraaf bij te houden. Uit iedere paragraaf wordt een ideaal aantal zinnen gemarkeerd om nadien geparafraseerd te worden door GPT-3 of een *HuggingFace* taalmodel, afhankelijk van de keuze van de eindgebruiker.

transformeren de tekst op lexicaal en syntactisch niveau. Zij bekijken enkel de gekregen zin. Andere taalmodellen zijn eerder geneigd om extra tekst toe te voegen. Er kan niet achterhaald worden waarom dat deze extra tekst wordt meegegeven. BART-SC kan bijzaak behouden, terwijl SC sneller de neiging heeft om enkel de kernzaak te behouden in de vereenvoudigde tekst.

Bij de inference API van T1 moet er expliciet worden aangegeven om welke transformatie dit gaat door kernwoorden zoals 'simplify:'.

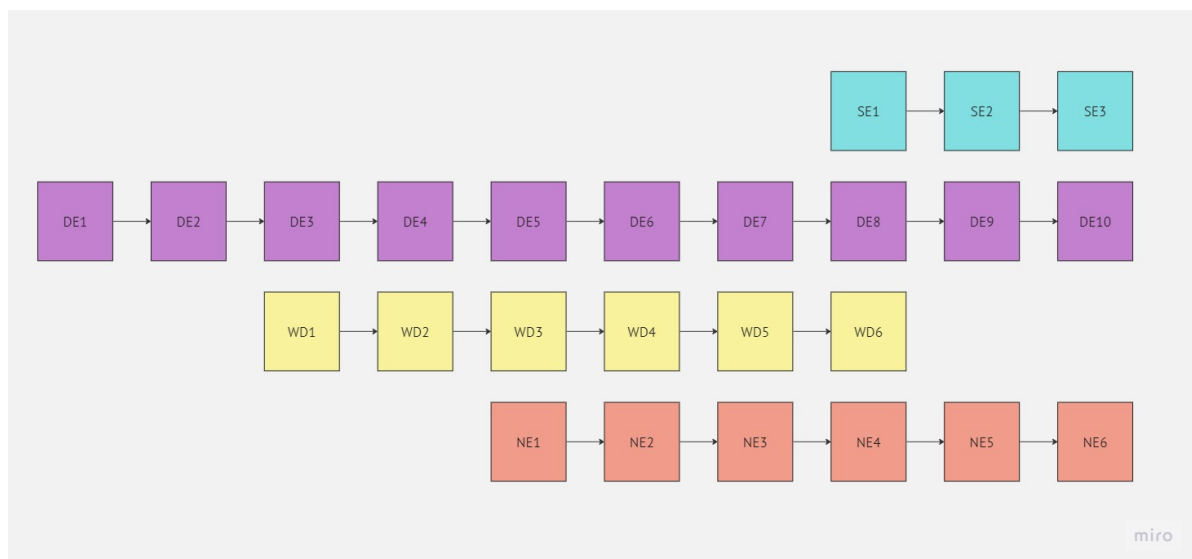
De zelfgebouwde Creator-klasse bouwt PDF's en docx-documenten op volgens de meegegeven personalisatie. Het prototype maakt gebruik van Pandoc, of Py-Pandoc via Python. Pandoc maakt gebruik van een tweestapsbeweging, waarbij plain-text eerst naar een Markdown-formaat wordt omgezet om vervolgens het Markdown-bestand naar een PDF of Word document te converteren.

Met Python wordt eerst een YAML-header in het te-transformeren Markdown-bestand geschreven. De YAML-header omvat de titel, standaardlettertype en lettertype voor de titel, de datum, het type document dat moet worden gegenereerd, de marge-instelling, de standaardlettergrootte, woord-spatiëring en ten slotte de instelling voor de regeleinde. De meegekregen gepersonaliseerde instellingen wordt meegegeven in een LaTeX YAML-header.

De structuur om de woordenlijst op te bouwen, is identiek zoals dat van een Markdown-

tabel. De woordenlijst wordt in dictionary-structuur meegegeven. De sleutels worden overlopen en vervolgens wordt ieder woord samen met de PoS-tag en de definitie uitgeprint. De vereenvoudigde tekst is eveneens in een dictionary-structuur opgeslaan. De keys stellen titels voor en worden uitgeprint voorafgegaan door twee hekje-symbolen. Tussen de titels worden breaklines toegevoegd, gevolgd door de tekst die bij de titel behoort. Indien gekozen werd voor een opsomming, dan wordt er gebruik gemaakt van een geneste for-lus waarbij iedere zin wordt voorafgegaan aan een asterisk-symbool. De woordenlijst en vereenvoudigde tekst worden naar hetzelfde Markdown-bestand uitgeschreven.

Als invoer wordt het pad naar opgevulde Markdown-bestand meegegeven. De uitvoer is het pad waarnaar het PDF- of DOCX-bestand moet worden opgeslaan. Vervolgens zet Pandoc het Markdown-bestand om naar een PDF-bestand gebouwd met de XeLateX engine of een Word-bestand op basis van meegekregen binaries. Pandoc Flask kan enkel één bestand aan de gebruiker teruggeven. Als oplossing comprimeert het prototype met *zipfile* de PDF- en Wordbestand tot één bestand.



Figuur (3.3)

Stappenplan voor de ontwikkeling van het component voor lectoren.

WD1	Flask-skelet aanmaken
WD2	Formulier voor GPT-3 API-sleutel invoer maken + sessie
WD3	Formulier voor gepersonaliseerde opties van website aanmaken + sessie
WD4	Webpagina's aanmaken in HTML & CSS
WD5	Invoerformulier maken voor PDF- en tekstupload
WD6	Invoerformulier maken voor het genereren van een gepersonaliseerde vereenvoudiging van een wetenschappelijk artikel

Tabel 3.10: Taken van de webontwikkelaar bij het uitwerken van het lerarencomponent.

NE1	Spacy word embeddings laden & PoS-tagging en lemmatization implementeren
NE2	Dictionary implementen voor het bijhouden van de PoS-tag per dictionary
NE3	Jupyter notebook om gepersonaliseerde prompts en aangepaste hyperparameters uit te testen voor de GPT-3 API
NE4	Jupyter notebook gebruiken om tekstvereenvoudigingsfuncties met GPT-3 API uit te testen.
NE5	Optioneel: Extra trainingsdata toevoegen aan GPT-3 model.
NE6	Code voor de voorgestelde pipeline voor ATS implementeren in Python back-end.

Tabel 3.11: Taken van de NLP Engineer bij het uitwerken van het lerarencomponent.

SE1	Dockerfile en bijhorende requirementsfile aanmaken
SE2	Opzet in Docker realiseren
SE3	Powershell en Bash-script realiseren

Tabel 3.13: Taken van de system engineer bij het uitwerken van het lerarencomponent.

DE1	Python-notebook om PDFMiner uit te testen bij willekeurige wetenschappelijke artikelen (2000 - nu)
DE2	Python-notebook opstellen om EasyOCR uit te testen bij willekeurige wetenschappelijke artikelen
DE3	Jupyter notebook om tekstdata cleaning te realiseren. De restanten van de PDF-extractie moeten weg.
DE4	Jupyter notebook om look-up methode voor synoniemen te realiseren.
DE5	Code in back-end implementeren voor PDF-upload via in-memory PDF read.
DE6	Python-notebook om Pandoc PDF & Word-document te genereren.
DE7	Uittesten van YAML-header in Markdown-bestand voor een document op maat.
DE8	Uittesten van uitschrijven tekstinhoud naar Markdown-bestand
DE9	Implementatie code van Pandoc in Flask-framework
DE10	Code in back-end implementeren voor zippen & doorsturen naar eindgebruiker.

Tabel 3.12: Taken van data engineer bij het uitwerken van het lerarencomponent.

3.3.2. Tool voor scholieren

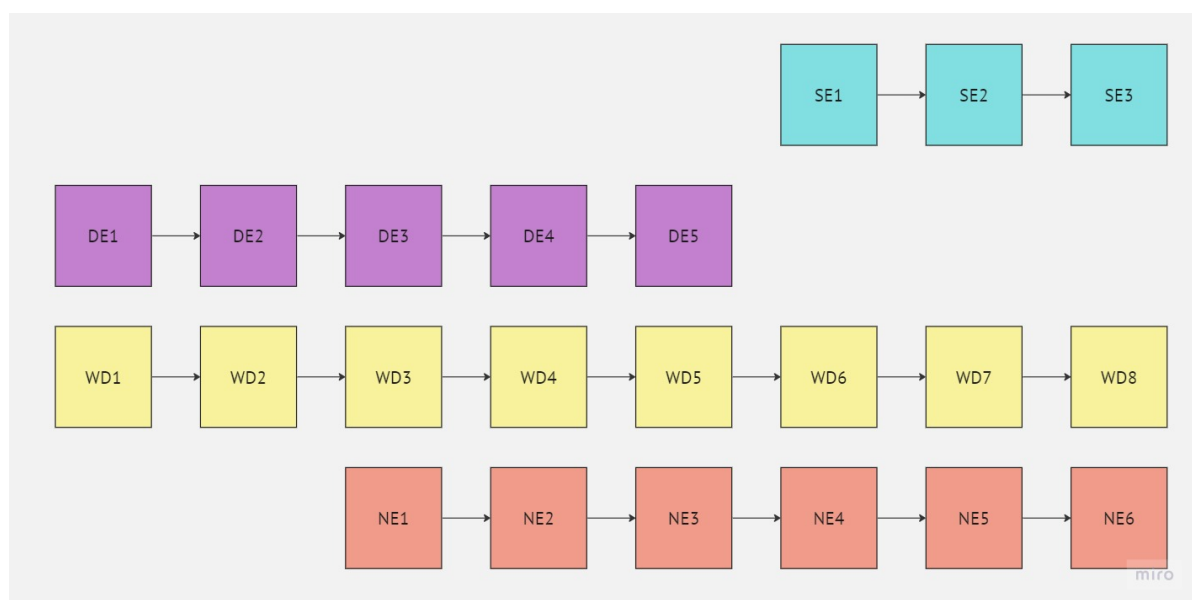
De taken van de NLP engineer blijven vrijwel dezelfde, dus hier wordt er verwezen naar de taken in tabel 3.11. De taken van de systeemingenieur vallen in dezelfde lijn als die bij het lerarencomponent, dus ook hier wordt er verwezen naar de taken beschreven in tabel ???. De bijhorende flowchart wordt weergegeven in figuur 3.4.

De weergave van de tool voor scholieren oogt gelijkaardig aan dat van de uitgeteste chatbots. Figuur ?? geeft een blik op hoe de webpagina er kan uitzien bij start. Met

Functionaliteit	JavaScript-functie	Python-functie
WD1	Flask-skelet aanmaken	
WD2	Formulier voor GPT-3 API-sleutel invoer maken + sessie	
WD3	Formulier voor gepersonaliseerde opties van website aanmaken + sessie	
WD4	Webpagina's aanmaken in HTML & CSS	
WD5	Invoerformulier maken voor PDF- en tekstupload	
WD6	JavaScript-functies schrijven voor het weergeven van grammaticale structuren, bijvoeglijke en zelfstandige naamwoorden.	
WD7	JavaScript functies schrijven voor dynamische tekstaanpassing met placeholder-tekst	
WD8	API-calls schrijven voor de functies: look-up, lexicale vereenvoudiging, formaatwijzigingen en ten slotte prompt-gedreven tekstvereenvoudiging	

Tabel 3.14: Taken van NLP engineer bij het uitwerken van het scholierencomponent.

JavaScript kunnen de volgende functionaliteiten aangeboden worden:



Figuur (3.4)

Stappenplan voor de ontwikkeling van het component voor scholieren.

DE1	Python-notebook om PDFMiner uit te testen bij willekeurige wetenschappelijke artikelen (2000 - nu)
DE2	Python-notebook opstellen om EasyOCR uit te testen bij willekeurige wetenschappelijke artikelen
DE3	Jupyter notebook om tekstdata cleaning te realiseren. De restanten van de PDF-extractie moeten weg.
DE4	Jupyter notebook om look-up methode voor synoniemen te realiseren.
DE5	Code in back-end implementeren voor PDF-upload via in-memory PDF read.

Tabel 3.15: Taken van data engineer bij het uitwerken van het scholierencomponent.

Ten slotte maakt het prototype gebruik van Docker voor de lokale opzet. Een scriptbestand, in Powershell of Bash, vereenvoudigt de opstart van deze webapplicatie, in tegenstelling tot de opstart per terminal. Alvorens installeert Docker met behulp van de voorafgegeven instructies de nodige Python-bibliotheken met het Pipreq-commando. Omdat het prototype gebruik maakt van de API's van de taalmodellen, hoeft het prototype geen gebruik te maken van een aparte container voor het taalmodel.

4

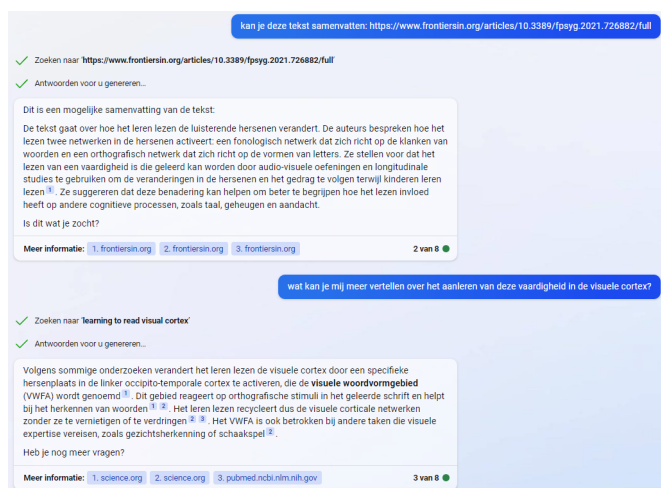
Resultaten

In dit hoofdstuk worden de resultaten uit de requirementsanalyse, vergelijkende studie en de ontwikkeling van het prototype besproken.

4.1. Requirementsanalyse

Woorden- en synoniemenlijsten kunnen een ondersteunend middel aanbieden voor zowel scholieren met dyslexie als zonder bij het lezen van wetenschappelijke artikelen en wordt aangeboden in Kurzweil. Automatisch genereren is enkel prevalent bij ChatGPT en de Bing chatbot, maar de tools houden geen rekening met de doelgroep, tenzij expliciet aangegeven met een *one-shot summary*. Andere tools houden helemaal geen rekening met de doelgroep en kunnen enkel woordenlijsten genereren op basis van gekozen woorden.

In figuur 4.1 wordt er een online wetenschappelijk artikel meegegeven. Er wordt geen titel of onderwerp meegegeven, maar de Bing AI chatbot is in staat om een abstraherende samenvatting te maken van het artikel. Daarna kan de chatbot meer uitleg geven over een bepaald onderwerp, alsook de referenties naar de oorspronkelijke bron meegeven.

**Figuur (4.1)**

Resultaat Bing Chatbot

Met de huidige erkende softwaretools in het onderwijs is het moeilijk om syntactische vereenvoudiging toe te passen op wetenschappelijke artikelen. Online webtoepassingen bieden weliswaar mogelijkheden om de moeilijkheidsgraad van de zinsstructuur te verlagen, maar ze zijn voornamelijk gericht op het verkorten van de oorspronkelijke tekst, of het maken van een samenvatting. Het aanpassen van tangconstructies, verwijswaarden, voorzetseluitdrukkingen, samengestelde werkwoorden en onregelmatige werkwoorden blijft daarom een uitdaging voor deze toepassingen. Zelfs het schrijven in de actieve stem kan problematisch zijn, en de beschikbare prompts zijn beperkt tot vooraf gedefinieerde transformaties.

Het uploaden van een PDF wordt beschouwd als een onmisbare optie bij de geteste tools, maar ChatGPT en Bing Chat beschikken niet over deze functie. De software die wordt gebruikt in het onderwijs kan geen tekst uit PDF's extraheren, in tegenstelling tot webtoepassingen die deze mogelijkheid wel bieden. De geteste tools zijn voornamelijk gericht op samenvatting van teksten. Hoewel ChatGPT en Bing Chatbot teksten kunnen genereren in een schrijfstijl vergelijkbaar met die van mensen, zijn ze niet in staat om PDF's te verwerken. Het kopiëren en plakken van tekst uit het originele document kan leiden tot fouten en is omslachtig, daarom moet er worden gewerkt aan verbetering van deze werkwijze. Bestaande tools hebben moeite met het extraheren van tekst uit oudere PDF's, dus het prototype moet een oplossing bieden voor dit mogelijke probleem.

Uitgezonderd van Simplish en Rewordify bieden de uitgeteste tools geen ingebouwde tekstanalysemodule en bieden onvoldoende inzicht op leesgraadsmetrieken van zowel het ingegeven document als het vereenvoudigde document. Simplish steekt boven de rest uit door aan de hand van kleurcodes criteria over de vereenvoudigde

tekst mee te geven, waaronder niet-veranderende woorden, adequate vertalingen, uitleg naar de voetnoot, homoniemen of woorden die geen eenvoudigere synoniemen hebben. Zoals aangegeven in 4.2 duidt de vergelijkende weergave de verschillen aan tussen de oorspronkelijke en vereenvoudigde tekst en met behulp van kleurcodes worden de verschillende transformaties aangegeven. Scholarcy vult hierop aan en biedt meer inzichten in de leesbaarheidsgraad van de vereenvoudigde tekst, zoals geïllustreerd in 4.3.

Color code	
Black	Words in Black don't change between the two versions.
Green	Words in Green mean they have been translated adequately.
Purple	Words in Purple display a further explanation in foot notes.
Olive	Words in Olive contain two or more possible meanings (a tooltip is provided for these words, place the mouse cursor on top of olive words to see possible meanings).
Blue	Words in Blue are recognized in Wikipedia (normally names, places, people, organizations, etc.).
Orange	Words in Orange are not currently available in Basic English.
Red	Words in Red are names, special terms or not recognized by the translating tool.

Note : Double click on any word to add it to your personal dictionary.

Input Text

Artificial intelligence has been applied more into occupations by companies and individuals. However, the effects within the benefits are both imaginable and unpredictable. Sexual discrimination in jobs is also a debatable topic. The purpose of this paper is to combine the topics of both AI and sexual discrimination and discuss their effects in the job field in the future. Automation, big data and the algorithm applied in the job field would be some of the points to discover. To briefly summarize, automation is the use of machines and computers that reduces human intervention. Big data is a collection of data from various sources, it is related to AI because the more data input into AI the better it becomes. Since AI absorbs the information and learns from them, AI algorithm takes the data input and uses mathematics and logic to produce the output. [1] Gender discrimination in AI not only reflects the pre-existing biases in the society, but it could also reinforce them through automation, hiring system and decision making. This paper is not totally against the use of AI but advocates that artificial intelligence should be used in a more careful, gender responsible way to reduce sexual discrimination in the job field.

Simplified

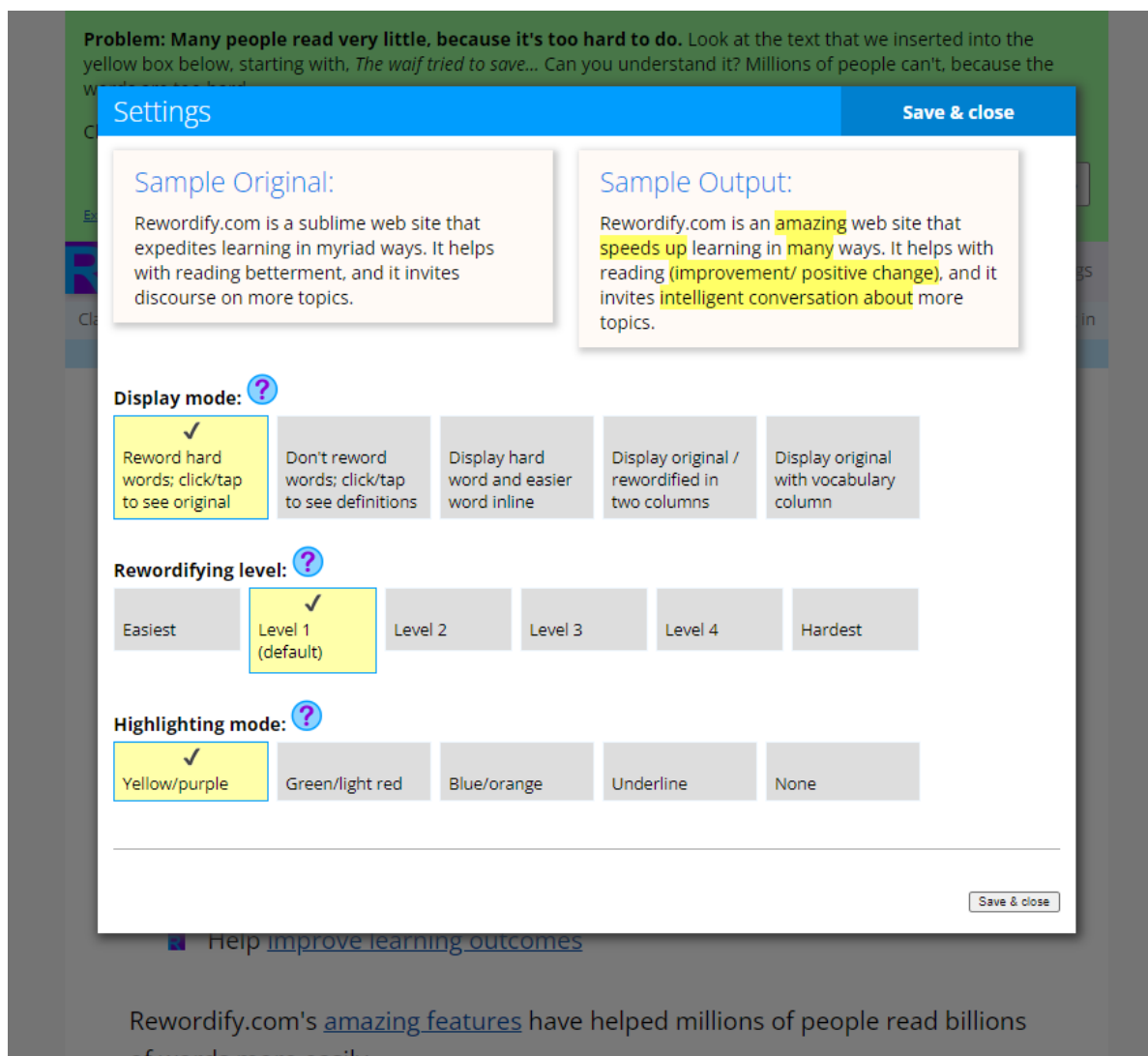
artificial intelligence ¹ has been made a request more into work by companies and beings. however, the effects within the gets help are both idea forming and not able to say before hand. sex caused decision making in regular work is also an about which argument is possible thing talked of. The purpose of this paper is to trading group the interests of both AI and sex caused decision making and have a discussion about their effects in the regular work field in the future. automation ², greatly sized facts and the algorithm ³ applied in the regular work field would be some of the points to discover. To briefly give a short account of, automation ² is the use of machines and knowledge processing machines that gets changed to other form to do with man coming between groups. Big facts is a group of facts from different starting points, it is related to AI because the more facts input into AI the better it becomes. Since AI takes up the news given and learns from them, AI algorithm ³ takes the facts input and uses mathematics and tests, reasoning to produce the out put. [1] sex statement decision making in AI not only gives back (light, heat, sound) the in existence beforehand has a tendency in a certain direction in the society, but it could also make stronger them through automation ², getting use of person for money system and decision making. This paper is not totally against the use of AI but Advocates ⁴ that artificial intelligence ¹ should be used in a more careful, sex statement responsible way to get changed to other form sex caused decision making in the regular work field.

Menu ▾

artificial intelli...	science that gives great weight to ways of making come into existence intelligent machines that work and have reactions like those of man. Continue reading.
automation ²	the technology of making machines, instruments, process, and the like go through a certain train of operations without further impulse or control from outside after being started. Continue reading.
algorithm ³	a word coming from the name of the expert in mathematics /Al-Khwarizmi@who (780-850ac), used to give the way to work out or solve points to be answered. Continue reading.
Advocates ⁴	A barrister or solicitor representing a party in a hearing before a Court. Continue reading.

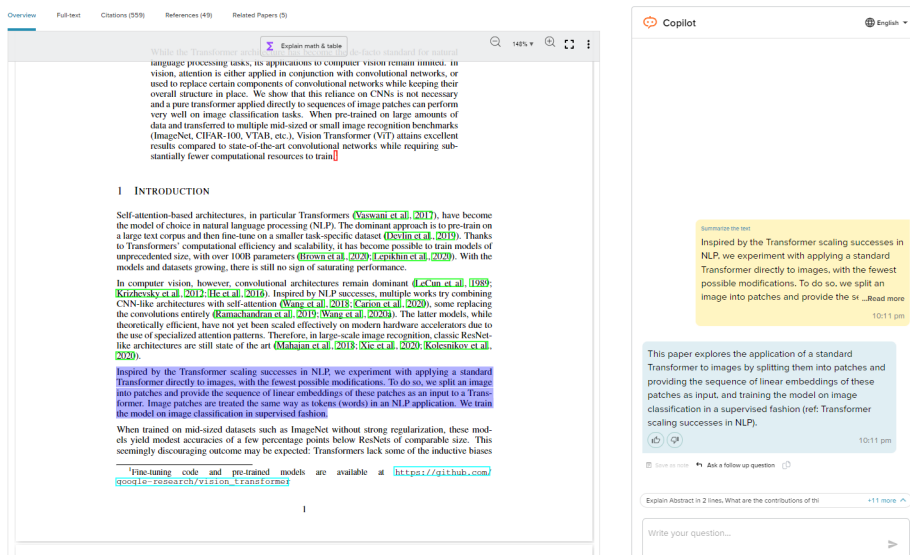
Figuur (4.2)

Illustratie van de tekstanalyse bij Simplish na een tekstvereenvoudiging.

**Figuur (4.3)**

Illustratie van de tekstanalyse bij Rewordify.

De requirementsanalyse neemt geen functionaliteiten op van obscure proof-of-concepten. Deze ontbreken de nodige testen en zekerheid dat deze effectieve gepersonaliseerde en geautomatiseerde tekstvereenvoudiging aanbieden. Twee van de online tools, namelijk Resoomer en Scispace, bieden uitsluitend samenvattingsfunctionaliteiten aan. Resoomer is sneller geneigd om de belangrijkste zinnen te markeren en vervolgens deze in een kortere tekst terug te geven. Dit kan een effect hebben op de betrouwbaarheid van de functionaliteiten, alsook op de leesgraadmetriecken.



Figuur (4.4)
Schermabbeelding van SciSpace

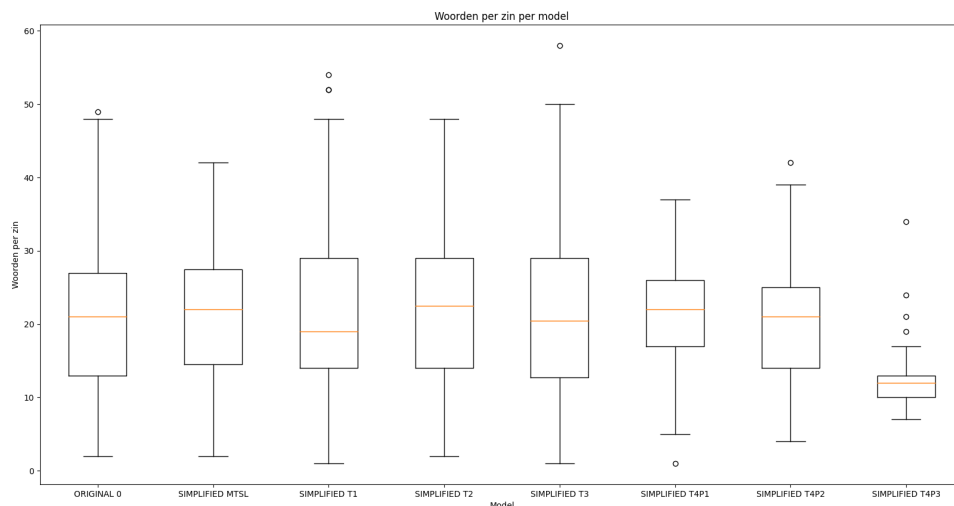
4.2. Vergelijkende studie

Na een tekstvereenvoudiging ligt het aantal zinnen bij T1, T2 en T3 hoger dan dat van de oorspronkelijke tekst. Elke prompt van T4 maakt gebruik van meer zinnen dan het oorspronkelijk artikel, zoals aangewezen in tabel ??.

Model	Zinnen in A1	Zinnen in A2
Oorspronkelijk	178	159
MTS (door leerkracht)	43	45
MTS (door leerling)	n.v.t.	50
T1	101	209
T2	82	209
T3	100	209
T4 P1	61	98
T4 P2	89	133
T4 P3	39	55

Tabel 4.1: Aantal zinnen

T1, T2 en T3 gebruiken gemiddeld minder woorden per zin dan het oorspronkelijk artikel. Enkel P3 is in staat om gemiddeld minder woorden per zin te gebruiken,

**Figuur (4.5)**

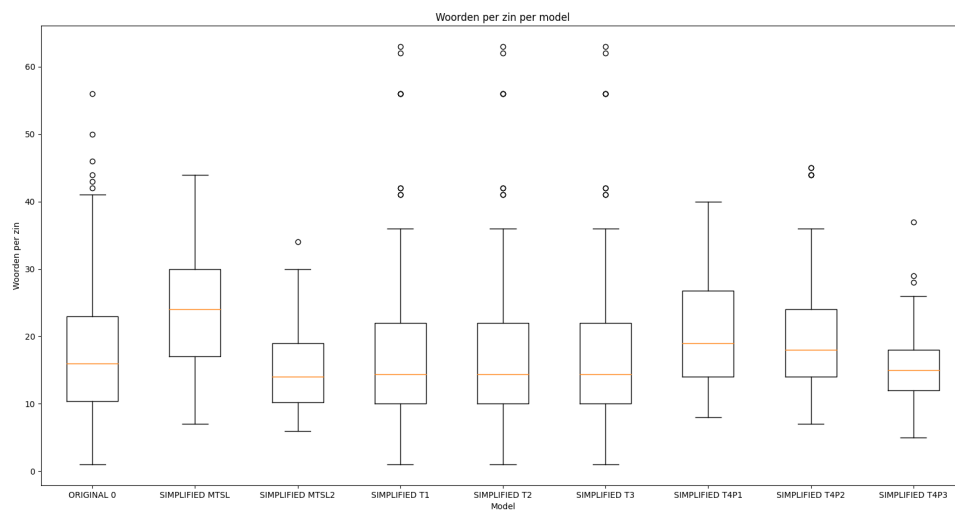
Overzicht van het minimum, maximum en gemiddeld aantal woorden per zin per model in artikel 1.

vergeleken met P1 en P2 die elk gemiddeld meer dan 19 woorden per zin gebruiken, zoals aangegeven in figuren 4.5 en 4.6.

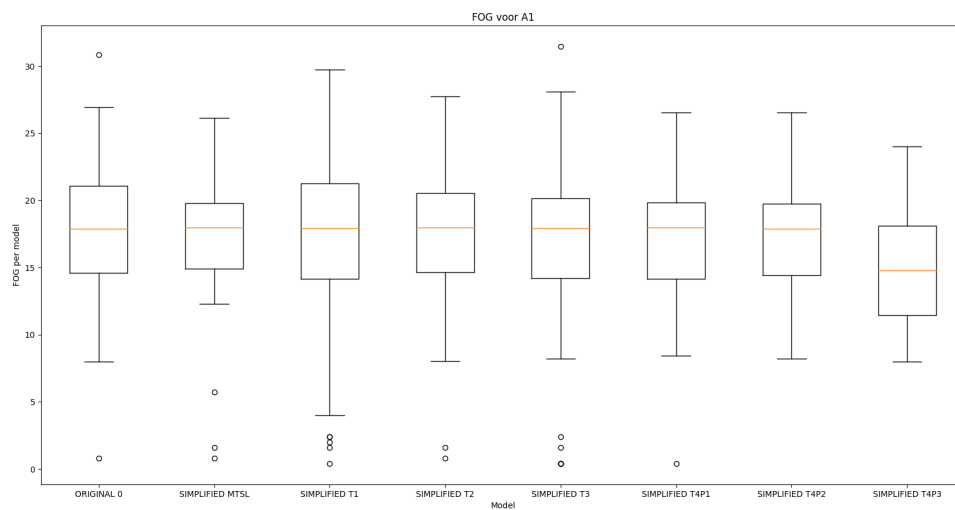
De FRE en FOG bij alle geteste taalmodellen en MTS-referentieteksten zijn niet significant hoger of lager, vergeleken met het oorspronkelijk wetenschappelijk artikel, zoals aangetoond in figuren 4.7, 4.8, 4.9 en 4.10.

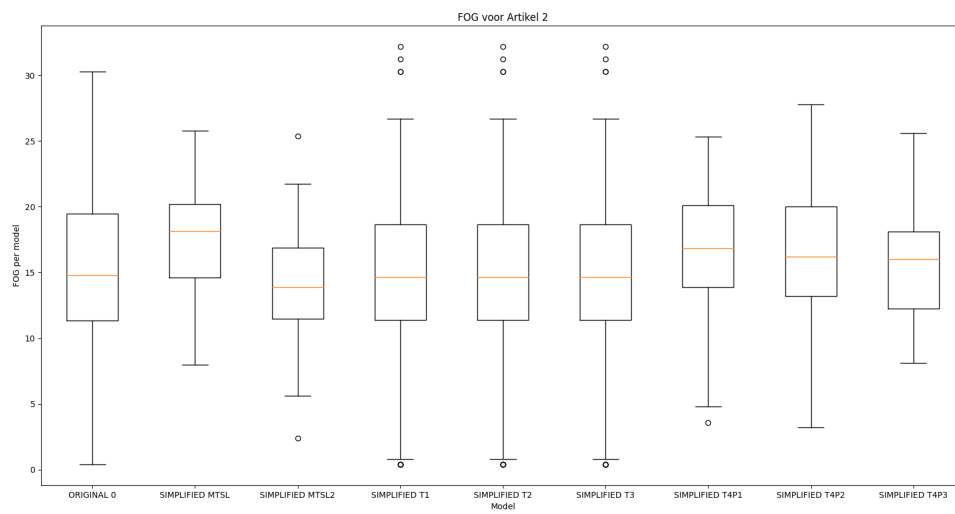
Modellen T1 en T2 zijn niet in staat om syntactische vereenvoudiging op een tekst toe te passen. T3 is bij alle prompts in staat om de syntax van een zin te verlagen. Daarnaast zijn er Alle uitgeteste taalmodellen zijn echter wel in staat om lexicale vereenvoudiging te realiseren, al wordt de inschatting van de doelgroep in twijfel getrokken. De referentieteksten schatten de doelgroep correct in, door reeds gekend jargon niet aan te passen, maar nieuwe jargon wel aan te passen naargelang er een beschikbaar synoniem is.

De modellen T1, T2 en T3 zijn niet in staat om syntactische vereenvoudiging op een tekst toe te passen. Alleen T4 kan via de prompts P2, P3, P4, P5 en P6 de zinsstructuur verlagen. Hoewel alle geteste taalmodellen in staat zijn om lexicale vereenvoudiging te realiseren, wordt de nauwkeurigheid van de doelgroepsinschatting in twijfel getrokken. De referentieteksten schatten de doelgroep correct in door bekend jargon niet aan te passen, maar wel nieuwe jargon aan te passen als er een beschikbaar synoniem is. Daarnaast kan T3-P1 ook de coherentie van een meegegeven paragraaf bevorderen, door onder meer omslachtige zinsstructuren aan te passen naar signaalwoorden.

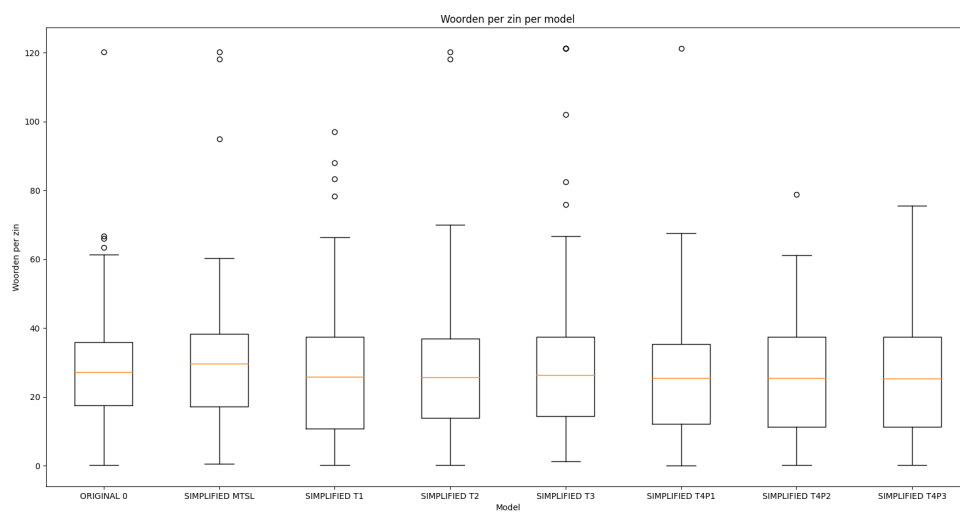
**Figuur (4.6)**

Overzicht van het minimum, maximum en gemiddeld aantal woorden per zin per model in artikel 2.

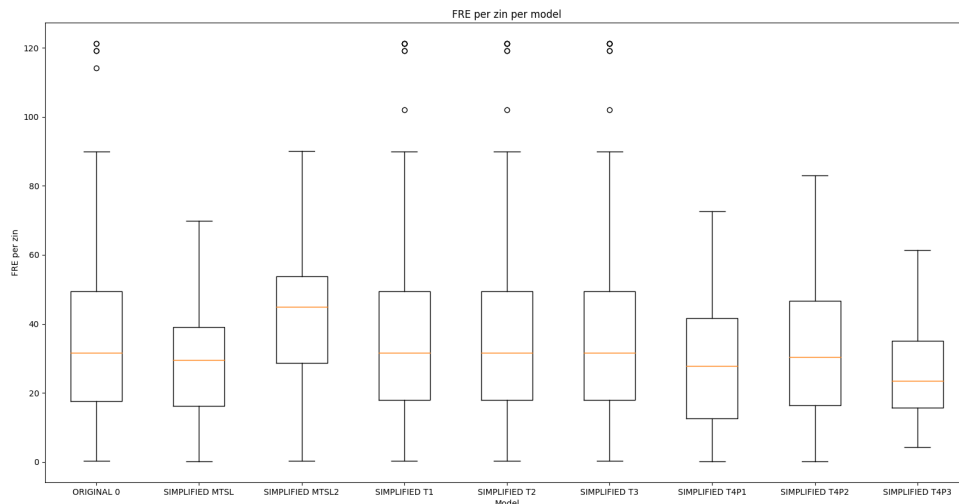
**Figuur (4.7)**



Figuur (4.8)



Figuur (4.9)



Figuur (4.10)

In tegenstelling tot GPT-3 zijn de modellen T1, T2 en T3 niet in staat om het formaat van de uitvoer aan te passen. De uitvoer blijft een doorlopende tekst. In de referentietekst past één van de auteurs het formaat aan naar tabelvorm voor enkele paragrafen, waar de inhoud beter in tabelvorm kan gestructureerd worden. Alleen bij de prompts P5 en P6 wordt er expliciet gevraagd om een formaatwijziging, anders geeft T4 vrijwel altijd een doorlopende tekst terug. Zonder de expliciete aanduiding is het model niet in staat om dit zelf te bepalen. Verder onderzoek moet uitwijzen of T4 in staat is om zelfstandig deze bepaling kan maken, zo niet kan deze formaatwijziging niet automatisch bepaald worden en is er tussenkomst van de eindgebruiker vereist. Dit moet ook zo opgenomen worden als een functionaliteit in het prototype.

Het formaat van de uitvoer bij de modellen T1 en T2 zijn identiek aan dat van de oorspronkelijke tekst, namelijk in de vorm van een doorlopende tekst. Deze twee modellen zijn niet in staat om het voorbeeld van één van de auteurs te volgen, door het formaat van de

De *execution time* bij zowel T1 en T2 scoren ondermaats, zowel via *inference API* als lokaal draaiend. Dit kan oplopen tot hoogstens 30 seconden voor een zin van tien tot dertig woorden, vergeleken met T3 die dezelfde zin in minder dan 10 seconden kan vereenvoudigen.

Hoewel T3 in staat is om zinsyntaxtransformaties uit te voeren, ervaart deze problemen ondervinden bij het verwerken van alle meegegeven transformaties. Het taalmodel kan transformaties ontbreken als alle transformaties in één prompt zijn betrokken. Een voorgestelde aanpak om dit tegen te gaan, bestaat uit een pipeline

van hoogstens drie prompts die elkaar opvolgen, zoals geïllustreerd in figuur ...

4.3. Opbouw van het prototype

Het prototype voldoet aan alle functionaliteiten die zijn gespecificeerd in het Moscow-schema **??**. Het overtreft daarmee elke andere tool uit de requirementsanalyse op alle gebieden. Het is vooral op het gebied van formaatwijzigingen de beste, wat suggereert dat het prototype de huidige staat van deze toepassingen weergeeft, gezien de consistente ontwikkeling van de personalisatieopties. Op het gebied van tekstvereenvoudiging scoort het prototype vergelijkbaar of iets beter dan wat GPT-3 kan, wat voor zich spreekt aangezien het twee identieke taalmodellen zijn.

Ontwikkelaars kunnen de opgebouwde flowchart volgen om team van vier rollen, namelijk systeem, data, NLP en web ontwikkelaars, te begeleiden doorheen de ontwikkeling van een toepassing voor tekstvereenvoudiging. De flowchart benadrukt dat deze handelingen ook perfect parallel kunnen worden uitgevoerd. Jupyter notebooks bieden een ontwikkelaarsvriendelijke manier om de code vooraf met een duidelijke visuele feedback te kunnen testen.

Dit prototype maakt geen gebruik van lokaal gehoste taalmodellen. Zo vermindert de nodige rekenkracht en geheugenruimte op het systeem waarop het prototype draait in vergelijking met een lokaal gehost taalmodel. Wanneer ontwikkelaars de toepassing willen uitrollen naar het grote publiek, kunnen ze overschakelen naar lokaal gehoste taalmodellen in plaats van taalmodellen per API. Door de taalmodellen verder te finetunen en te trainen op meer datasets, kunnen AI-ontwikkelaars betere resultaten behalen.

Het ontwikkelen van gebruikersvriendelijke handelingen kan gemakkelijk en snel worden ontwikkeld met behulp van HTML, CSS en JavaScript, zonder de nood van een complex framework. Uit onderzoek blijkt echter dat deze methoden eenvoudig te ontwikkelen zijn, zelfs door pas afgestudeerde bachelorstudenten. AI-softwarebedrijven zouden meer moeten inzetten op personalisatie-opties voor scholieren met dyslexie in de derde graad van het middelbaar onderwijs, omdat zij momenteel het meest betrokken zijn bij de digitalisering. Later kunnen softwareontwikkelaars overgaan op een complexer of robuuster framework. Verder onderzoek is nodig naar het ideale framework om een gepersonaliseerde weergave mogelijk te maken, aangezien dit momenteel nog wordt gedaan met handmatige JavaScript-functies. Een front-end framework dat dit alles beheert, zou handiger zijn voor webontwikkelaars.

Dit prototype is momenteel alleen beschikbaar in een lokale omgeving en kan nog

niet worden gebruikt door het grote publiek. Het prototype kan teksten lexicaal en syntactisch vereenvoudigen wanneer deze in PDF- of volledige tekstformaat worden ingevoerd. Het prototype heeft functionaliteit voor zowel docenten als studenten, die elk verschillende prioriteiten hebben. Hoewel testen nog nodig zijn, wordt dit wel aangeraden. Onderzoekers op het gebied van logopedie kunnen het prototype testen bij studenten met dyslexie na een (begeleide) installatie van de vereiste set-up. Ook onderzoekers op het gebied van onderwijs kunnen dit gebruiken om de meningen van zowel studenten als docenten te verzamelen en zo de effectiviteit van het prototype te beoordelen. Deze experimenten zijn belangrijk omdat ze kunnen wijzen op de effectiviteit van het prototype.

5

Conclusie

In deze scriptie is gezocht naar een antwoord op de volgende onderzoeksvraag:

- Hoe kan een wetenschappelijke artikel automatisch vereenvoudigd worden, gericht op de unieke noden van scholieren met dyslexie in de derde graad middelbaar onderwijs?

Uit de requirementsanalyse werden nieuwe inzichten gegeven omtrent het bestand toepassen van ATS. Zo beschikken online tools echter te weinig over gepersonaliseerde ATS-functionaliteiten, zoals gebleken in sectie 3.1. Daarnaast maken weinig tools gebruik van gepersonaliseerde weergaveopties, die echter een bewezen effect hebben op het leesbegrip van een scholier met dyslexie in de derde graad van het middelbaar onderwijs.

Uit de vergelijkende studie blijkt dat GPT-3 de leesbaarheid van een wetenschappelijke tekst, gemeten met FRE en Gunning FOG, kan verbeteren door middel van gepersonaliseerde ATS. Andere geteste taalmodellen uit HuggingFace behalen minder accurate resultaten en vereisen een extra vertaalfase, die redundant is bij het aanspreken van de GPT-3 API.

Uit de ontwikkeling van het prototype voor gepersonaliseerde ATS is gebleken dat open-source AI- en NLP-technologieën hoogstaand genoeg zijn om kwaliteitsvolle tekstvereenvoudigingssoftware te ontwikkelen. Zo kunnen ontwikkelaars gebruik maken van PDFMiner om tekstinhoud uit wetenschappelijke artikelen te extraheren, OpenAI's GPT-3 model via de API om gepersonaliseerde ATS mogelijk te maken en ten slotte Pandoc om dynamische en gepersonaliseerde PDF-documenten automatisch te genereren. Binnen een webapplicatie kunnen complexe commandline-

handelingen afgehandeld worden door eenduidige handelingen gebouwd in Javascript en HTML&CSS.

6

Discussie

Voor dit onderzoek zijn vier verschillende onderzoeksmethoden toegepast om te bepalen hoe ontwikkelaars een optimale vorm van gepersonaliseerde ATS kunnen bieden aan scholieren met dyslexie in de derde graad van het middelbaar onderwijs.

Afhankelijk van de voortgang van individuele toepassingen, zouden de resultaten van de requirementsanalyse consistent zijn. Uit de resultaten van de requirementsanalyse blijkt dat zowel erkende toepassingen als online tools zonder chatbot weinig functionaliteiten bieden voor gepersonaliseerde ATS. De twee geteste GPT-3 tools, namelijk ChatGPT en Bing Chatbot, bieden wel deze functionaliteiten, maar missen intuïtieve handelingen waardoor leeks er moeilijk mee kunnen werken. Dit resultaat komt overeen met de verwachting dat bestaande tools niet gespecialiseerd zijn in gepersonaliseerde ATS op maat van scholieren met dyslexie in de derde graad van het middelbaar onderwijs. Een mogelijke verklaring voor dit resultaat is de populariteit aan samenvattingstools vergeleken met vereenvoudigingstools. Daarnaast bieden Bing Chatbot en ChatGPT enkel een conversationele chatbot aan, zonder de focus op een tool bedoeld voor gepersonaliseerde ATS.

Hetzelfde geldt voor de vergelijkende studie, waarbij de Huggingface-taalmodellen vergelijkbare tot identieke resultaten zouden behalen. Uit de vergelijkende studie van taalmodellen voor gepersonaliseerde ATS bleek zowel de drie uitgeteste HuggingFace-modellen via API, alsook het uitgeteste GPT-3 model via API, in staat zijn om moeilijke woorden te achterhalen en te vervangen. HuggingFace-modellen zijn echter beperkt en vereisen een vertaling naar het Engels, vergeleken met het GPT-3 model dat capabel is om gepersonaliseerde ATS aan te bieden zonder verplichte vertaling. Dit resultaat is eveneens in overeenstemming met de verwachting dat GPT-3 beter capabel is in om gepersonaliseerde ATS aan te bieden, vergeleken met vrij beschikbare HuggingFace taalmodellen. Een mogelijke verkla-

ring voor dit resultaat is de data waarop deze taalmodellen zijn getraind, alsook de complexiteit van de taalmodellen mede door het aantal parameters waaruit deze bestaan. De drie HuggingFace-taalmodellen zijn voornamelijk getraind op wetenschappelijke data, maar niet specifiek op wetenschappelijke artikelen die met MTS of ATS vereenvoudigd zijn voor 16 tot 18-jarigen met dyslexie in het middelbaar onderwijs. GPT-3 beschikt over een stuk meer parameters en kan daarmee ook meer complexere taalverwerkingen behandelen.

Er is meer onderzoek nodig om de bruikbaarheid van deze scores te bepalen en om te begrijpen hoe de scores zich verhouden tot de kwaliteit van de vereenvoudigde tekst. De manuele vergelijking met referentieteksten, vereenvoudigd met MTS, biedt een inkijk in hoe lectoren teksten kunnen vereenvoudigen. De vergelijkende studie hield geen rekening met het hoofdstuk waarin een zin werd beoordeeld. Vragen naar het verschil na een tekstvereenvoudiging per hoofdstuk in een wetenschappelijk artikel kan daarmee niet beantwoord worden en moet opgevolgd worden in een verder onderzoek.

Ontwikkelaars kunnen de stappen in het stappenplan volgen om een gelijksoortig resultaat te bereiken. Uit de ontwikkeling van het prototype bleek dat ontwikkelaars met vrij beschikbare middelen en API's in staat zijn om gepersonaliseerde ATS aan te reiken aan scholieren met dyslexie in de derde graad van het middelbaar onderwijs. Dit resultaat is in overeenstemming met de verwachting dat ontwikkelaars over de nodige tools beschikken om een dergelijk prototype voor gepersonaliseerde ATS te kunnen realiseren. Een mogelijke verklaring voor dit resultaat is de beschikbaarheid aan open-source tools en Python-libraries die complexe opdrachten, zoals het genereren van dynamische PDF- of Word-documenten, eenvoudig kunnen aanbieden aan ontwikkelaars.

Aan het resultaat ligt mogelijk ook ten grondslag dat ...

Hoewel deze scores een goed alternatief zijn om de leesbaarheid te meten, houden ze geen rekening met verkeerd geïnterpreteerde resultaten, zoals letterlijk overgenomen bronvermeldingen of verkeerd gegenereerde woordenschat door het taalmodel. Deze fouten zijn minder vaak aanwezig bij het vereenvoudigen van teksten met GPT-3. Hoewel vrij beschikbare modellen op HuggingFace in staat zijn om lexicale vereenvoudiging mogelijk te maken, staan ze in de schaduw van GPT-3, dat voor ontwikkelaars vrij beschikbaar is in de vorm van een API. Reeds ge-finetuned taalmodellen, beschikbaar op HuggingFace, bieden een oplossing voor niet-gepersonaliseerde ATS. Voor gepersonaliseerde ATS, bestaande uit specifieke lexicale en syntactische vereenvoudigingstaken, zijn LLM's zoals GPT-3 geschikt. Ontwikkelaars moeten echter rekening houden met de schaal van de modellen bij het maken van deze keuze.

Functionaliteiten combineren is een haalbare zaak voor zowel klein- als grootschalige softwareondernemingen. Het prototype is opgebouwd uit kennis en tools die

aangeleerd worden in alle richtingen Toegepaste Informatica bij alle Vlaamse hogescholen. Met gebruik van kant-en-klare taalmodellen, API's en gekende programmeertalen zijn ontwikkelaars ertoe in staat om een webtoepassing te ontwikkelen die ondersteuning kan bieden aan scholieren met dyslexie in de derde graad van het middelbaar onderwijs.

Het is voorbij aan het doel van deze studie om specifieke handelingen bij het uitvoeren van de acties bij een tool voor tekstvereenvoudiging te onderzoeken. Enkel de toegepaste tekstvereenvoudigingsfunctionaliteiten in de tools komen aan bod. De studie kan niet het opvolgende GPT-model, namelijk GPT-4, in de vergelijkende studie opnemen. Dit model is echter enkel tegen betaling beschikbaar, daarnaast is er ook geen bewezen onderzoek naar het gebruik van dit LLM.

De lezer moet rekening houden dat de ontwikkeling van de presentatie van de webtool gebaseerd is op eerder onderzochte visuele kenmerken, alsook op onderzoeken die eerder de impact van tekstvereenvoudiging met MTS aanwezen bij scholieren met dyslexie tijdens het intensief lezen van algemene teksten. Daarmee is het prototype niet uitgetest bij het doelpubliek tijdens het intensief lezen van een wetenschappelijk artikel en fungeert daarmee enkel als meting van de haalbaarheid voor ontwikkelaars bij de ontwikkeling van een dergelijke tool.

Bestaande taalmodellen vereenvoudigen de ontwikkeling van toepassingen op het gebied van semantische analyse, kernwoordenidentificatie en het extraheren van samenvattingen.

LLM's, waaronder GPT-3, zijn in staat om vragen te beantwoorden. Een mogelijke toevoeging aan dit prototype is het kunnen stellen van vragen omtrent de inhoud. Eerder onderzoek rond GPT-3 heeft uitgewezen dat het model in staat is om een tekst door te nemen, en deze vervolgens te gebruiken als kennis om vragen te beantwoorden. De tekstinhoud van een wetenschappelijk artikel wordt niet bijgehouden in dit prototype, maar een volgend onderzoek dat hier wel rekening mee houdt kan wederom een revolutionaire oplossing bieden in het kader van ondersteunend intensief lezen.

Het prototype is ontwikkeld met een snelle en intuïtieve deployment in gedachten. Door het meegegeven script-bestand is enkel de installatie van Docker Desktop vereist. Al is een online deployment optimaler, voor experimentdoeleinden is deze opzet ideaal en met een intuïtieve handeling kunnen leeks ook van dit prototype gebruikmaken, inclusief met de gegeven instructies.

- Onderzoekers binnen het vakdomein logopedie kunnen dit prototype gebruiken om experimenten af te nemen die het effect op leesbegrip achterhalen bij scholieren met dyslexie in de derde graad van het middelbaar onderwijs. Er ontbreekt wetenschappelijke vakliteratuur in het kader van tekstvereenvoudiging met ATS voor deze specifieke doelgroep, dus wetenschappelijke experimenten kunnen revolutionaire inzichten aanreiken.

- Onderzoekers binnen het vakdomein secundair onderwijs kunnen de effecten bij leerlingen en leerkrachten in het middelbaar onderwijs waarnemen, alsook besluiten of deze tool al dan niet van pas kan komen.

Alhoewel GPT-3 een overkill is voor NLP-taken, waaronder kernwoordherkenning of extraherende samenvatting, die kosteneffectief kunnen aangepakt worden zonder het gebruik van GPT-3. Een LLM hoeft niet voor iedere functionaliteit ingezet te worden, zo om kostenbesparend te werken. Het GPT-3 model maakt complexe en granulaire NLP-transformaties op lexicaal en syntactisch niveau mogelijk voor gepersonaliseerde ATS. Echter houdt het model geen rekening met referenties buiten de getrainde data, wat tot problemen bij de *data integrity* kan leiden. Bing AI daarentegen doet dit wel en vormt een goede fundering voor ontwikkelaars om meer referentiemateriaal aan te bieden in ondersteunende software binnen het onderwijs. Verder onderzoek op de toepassing van deze AI via een mogelijke API is zeker nodig en kan baanbrekend zijn voor de onderwijssector.

Het toegepaste GPT-3 model is enkel gefinetuned per API-parameters en bevat geen vooraf getrainde extra data van wetenschappelijke papers. Er is echter wetenschappelijke data beschikbaar die kan worden gebruikt om het GPT-3 taalmodel accurater te maken op interpretatie van complexiteit bij wetenschappelijke artikelen. Er is een licht effect waargenomen op de verschillen in lexicale complexiteit tussen de HuggingFace-taalmodellen die wel getraind zijn op wetenschappelijke papers in vergelijking met taalmodellen die getraind zijn op algemene data, maar meer onderzoek is nodig om deze verschillen beter te begrijpen binnen de context van wetenschappelijke papers. Het is belangrijk op te merken dat de taalmodellen van OpenAI voortdurend evolueren en dat er overwogen wordt om GPT-2 achterwege te laten in het licht van verdere edities van de GPT-modellen. Op dit moment worden GPT-4 en Bing AI uitgerold, maar deze zijn nog niet klaar voor gebruik in productie, dus verder onderzoek is nodig naar het gebruik van deze modellen in het onderwijs. GPT-3 kan een baanbrekende oplossing aanbieden voor geautomatiseerde tekstvereenvoudiging van wetenschappelijke artikelen.

- Het gebruik van GPT-3 maakt het mogelijk om moeilijke woorden snel en efficiënt te identificeren binnen een doorlopende tekst. Toekomstig onderzoek zou zich kunnen richten op het potentieel van de combinatie van GPT-3 en full-text-search technologieën, waarbij specifieke zoektermen en thema's worden gebruikt om woordenlijsten te genereren. Dit zou kunnen bijdragen aan een nog meer geoptimaliseerde ondersteuning van het leerproces.

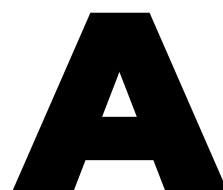
Met alsmaar grotere taalmodellen, zoals het opkomende GPT-4 en LLaMa, is er ook meer onderzoek nodig naar de verschillen op taalvlak ten opzichte van de toename in parameters. Het GPT-3 model dat in dit onderzoek werd gebruikt, maakte enkel gebruik van aangepaste parameters zoals de *temperature* en *top_p*. Hoewel de

overstap naar andere taalmodellen kostelijk kan zijn voor ontwikkelaars, is het belangrijk om te onderzoeken of en hoe deze nieuwe modellen kunnen bijdragen tot betere resultaten. Het is echter eerder uitgewezen dat de grootte van taalmodellen alsmear minder relevant wordt.

De erkende software uitgeleend aan scholieren met dyslexie in de derde graad van het middelbaar onderwijs voldoet niet aan de noden. De software biedt ondersteunende functionaliteiten aan zoals het aanmaken van een woordenlijst, alsook het markeren van zinnen om deze later om te vormen naar een tekst. Syntactische vereenvoudiging of abstraherende samenvatting zijn niet tot de orde. Online toepassingen staan verder en reiken functionaliteiten aan die hoogstaand zijn en reproduceerbaar zijn voor mensen met informaticakennis. Echter is er geen manusje-van-alles en er is daarmee nood aan een toepassing die alle functionaliteiten kan combineren. De erkende softwarepakketten zoals Kurzweil kunnen opgeschaald worden, zodat deze de functionaliteiten hebben om verbeterde tekstvereenvoudigingstechnieken aan te reiken aan scholieren met dyslexie in de derde graad van het middelbaar onderwijs.

Onderzoek naar het verschil tussen het laten schrijven van prompts en vooraf gedefinieerde prompts is schaars, maar deze keuze kan een effect hebben op het gedrag of ervaring van de eindgebruiker. De doelgroep wordt expliciet aangeduid in de prompts en is daarmee parameteriseerbaar. Er is verder onderzoek nodig naar de effecten op het meegeven van doelgroepen via prompts en of deze al dan niet rekening houden met de doelgroep.

Er is echter meer onderzoek nodig naar hoe de inzet van webtoepassingen en browserextensies voor tekstvereenvoudiging in het onderwijs kan worden verbeterd. Hero van Bingel e.a. (2018) biedt eveneens een browserextensie aan, maar dit is enkel voor selecte websites.



Onderzoeksvoorstel

Samenvatting

Ingewikkelde woordenschat en zinsbouw hinderen scholieren met dyslexie in het derde graad middelbaar onderwijs bij het lezen van wetenschappelijke artikelen. Gepersonaliseerde tekstvereenvoudiging helpt deze scholieren bij hun leesbegrip. Daarnaast kan artificiële intelligentie (AI) dit proces automatiseren om de werkdruk bij leraren en scholieren te verminderen. Dit onderzoek achterhaalt met welke technologische en logopedische aspecten AI-ontwikkelaars rekening moeten houden bij de ontwikkeling van een AI-toepassing voor geautomatiseerde en gepersonaliseerde tekstvereenvoudiging. Hiervoor is de volgende onderzoeksvraag opgesteld: "Hoe kan een wetenschappelijk artikel automatisch worden vereenvoudigd, gericht op de unieke noden van scholieren met dyslexie in het derde graad middelbaar onderwijs?". Een requirementsanalyse achterhaalt de benodigde functionaliteiten om gepersonaliseerde en geautomatiseerde tekstvereenvoudiging mogelijk te maken. Vervolgens wijst de vergelijkende studie uit welk taalmodel kan worden ingezet om de taak van gepersonaliseerde en geautomatiseerde tekstvereenvoudiging mogelijk te maken. De requirementsanalyse wijst uit dat toepassingen om wetenschappelijke artikelen te vereenvoudigen, gemaakt zijn voor een centrale doelgroep en geen rekening houden met de unieke noden van een scholier met dyslexie in het derde graad middelbaar onderwijs. Adaptieve software voor geautomatiseerde tekstvereenvoudiging is mogelijk, maar ontwikkelaars moeten meer inzetten op de unieke noden van deze scholieren.

A.1. Introductie

Het Vlaams middelbaar onderwijs staat op barsten. Leraren en scholieren worden overspoeld door werkdruk en stress. Bovendien is de derde graad van het middelbaar onderwijs een belangrijke mijlpaal voor de verdere loopbaan van scholieren,

al hebben zij volgens Dapaah en Maenhout (2022) dan moeite om grip te krijgen op de vakliteratuur bij STEM-vakken. Het STEM-agenda¹ van de Vlaamse Overheid moet het STEM-onderwijs tegen 2030 aantrekkelijker te maken, door de ondersteuning voor zowel leerkrachten als scholieren te verbeteren. Toch wordt het aanpakken van de steeds complexere wetenschappelijke taal, zoals beschreven in Barnett en Doubleday (2020), niet opgenomen in het STEM-agenda. Wetenschappelijke artikelen vereenvoudigen, op maat van de noden voor een scholier met dyslexie in het middelbaar onderwijs, is tijds- en energie-intensief voor leerkrachten en scholieren. Automatische en adaptieve tekstvereenvoudiging biedt hier een baanbrekende oplossing om de werkdruk in het middelbaar onderwijs te verminderen.

Het doel van dit onderzoek is om te achterhalen met welke technologische en logopedische aspecten AI-ontwikkelaars rekening moeten houden bij de ontwikkeling van een adaptieve AI-toepassing voor geautomatiseerde tekstvereenvoudiging. De volgende onderzoeksvraag is opgesteld: "Hoe kan een wetenschappelijke artikel automatisch vereenvoudigd worden, gericht op de verschillende behoeften van scholieren met dyslexie in de derde graad middelbaar onderwijs?". Het doel wordt bereikt door een antwoord op de volgende deelvragen te formuleren. Eerst geeft de literatuurstudie een antwoord op de eerste vier deelvragen. Daarna vormt het veldonderzoek een antwoord op de vijfde deelvraag. Ten slotte beantwoordt de vergelijkende studie de zesde en laatste deelvraag. De resultaten van dit onderzoek zetten AI-ontwikkelaars aan om een toepassing te maken om scholieren met dyslexie te kunnen ondersteunen in de derde graad middelbaar onderwijs.

1. Welke aanpakken zijn er voor geautomatiseerde tekstvereenvoudiging? Aansluitende vraag: "Hoe worden teksten handmatig vereenvoudigd voor scholieren met dyslexie?"
2. Welke specifieke noden hebben scholieren van de derde graad middelbaar onderwijs bij het begrijpen van complexere teksten?
3. Wat zijn de specifieke kenmerken van wetenschappelijke artikelen?
4. Met welke valkuilen bij taalverwerking met AI moeten ontwikkelaars rekening houden?
5. Welke toepassingen, tools en modellen zijn er beschikbaar om Nederlandstalige geautomatiseerde tekstvereenvoudiging met AI mogelijk te maken?
6. Welke functies ontbreken AI-toepassingen om geautomatiseerde én adaptieve tekstvereenvoudiging mogelijk te maken voor scholieren met dyslexie in de derde graad middelbaar onderwijs? Aansluitende vraag: "Welke manuele methoden voor tekstvereenvoudiging komen niet in deze tools voor?"

¹<https://www.vlaanderen.be/publicaties/stem-agenda-2030-stem-competenties-voor-een-toekomst-en-missiegericht-beleid>

A.2. State-of-the-art

A.2.1. Tekstvereenvoudiging

De voorbije tien jaar is artificiële intelligentie (AI) sterk verder ontwikkeld. Vasista (2022) benadrukt dat de toename in kennis voor nieuwe toepassingen zorgde. Tekstvereenvoudiging vloeide hier uit voort. Momenteel bestaan er al robuuste toepassingen die teksten kunnen vereenvoudigen, zoals Resoomer², Paraphraser³ en Prepostseo⁴. Binnen het kader van tekstvereenvoudiging is er bestaande documentatie beschikbaar waar onderzoekers het voordeel van toegankelijkheid aanhalen, maar volgens Gooding (2022) ontbreken deze toepassingen de extra noden die scholieren met dyslexie in de derde graad middelbaar onderwijs vereisen.

Shardlow (2014) haalt aan dat het algemene doel van tekstvereenvoudiging is om ingewikkelde bronnen toegankelijker te maken. Het zorgt voor verkorte teksten zonder de kernboodschap te verliezen. Siddharthan (2014) haalt verder aan dat tekstvereenvoudiging op één van drie manieren gebeurt. Er is conceptuele vereenvoudiging waarbij documenten naar een compacter formaat worden getransformeerd. Daarnaast is er uitgebreide modificatie die kernwoorden aanduidt door gebruik van redundantie. Als laatste is er samenvatting die documenten verandert in kortere teksten met alleen de topische zinnen. Met deze concepten zijn ontwikkelaars volgens Siddharthan (2014) in staat om ingewikkelde woorden te vervangen door eenvoudiger synoniemen of zinnen te verkorten zodat ze sneller leesbaar zijn.

Tekstvereenvoudiging behoort tot de zijtak van *Natural Language Processing* (NLP) in AI. NLP omvat methodes om menselijke teksten om te zetten in tekst voor machines. Documenten vereenvoudigen met NLP kan volgens Chowdhary (2020) op twee manieren: extract of abstract. Bij extractieve vereenvoudiging worden zinnen gelezen zoals ze zijn neergeschreven. Vervolgens bewaart een document de belangrijkste taalelementen om de tekst te kunnen hervormen. Deze vorm van tekstvereenvoudiging komt volgens (Sciforce, 2020) het meeste voor. Daarnaast is er abstracte vereenvoudiging waarbij de kernboodschap wordt bewaard. Met de kernboodschap wordt er een nieuwe zin opgebouwd. Volgens het onderzoek van Chowdhary (2020) heeft deze vorm potentieel, maar het zit nog in de kinderschoenen.

A.2.2. Noden van scholieren met dyslexie

Het experiment van Franse wetenschappers

Gala en Ziegler (2016) illustreert dat manuele tekstvereenvoudiging schoolteksten toegankelijker

maakt voor kinderen met dyslexie. Dit deden ze door simpelere synoniemen en

²<https://resoomer.com/nl/>

³<https://www.paraphraser.io/nl/tekst-samenvatting>

⁴<https://www.prepostseo.com/tool/nl/text-summarizer>

zinsstructuren te gebruiken. Tien kinderen werden opgenomen in het experiment, variërend van 8 tot 12 jaar oud. Verwijswoorden werden vermeden en woorden kort gehouden. De resultaten waren veelbelovend. Het leestempo lag hoger en de kinderen maakten minder leesfouten. Ook bleek er geen verlies van begrip in de tekst bij geteste kinderen. Resultaten van de studie werden gebundeld voor de mogelijke ontwikkeling van een AI-tool.

De visuele weergave van tekst beïnvloedt de leessnelheid bij scholieren met dyslexie. Zo haalt het onderzoek van Rello e.a. (2012) tips aan waarmee teksten en documenten rekening moeten houden bij scholieren met dyslexie in de derde graad middelbaar onderwijs. Het gaat over speciale lettertypes, spreiding tussen woorden en het gebruik van inzoomen op aparte zinnen. Het onderzoek haalt verder aan dat teksten voor deze unieke noden aanpassen tijdrovend is, dus tekstvereenvoudiging door AI kan een revolutionaire oplossing bieden. De Universiteit van Copenhagen is met bovenstaande idee aan de slag gegaan. Onderzoekers Bingel e.a. (2018) hebben gratis software ontwikkeld, genaamd Hero⁵, om tekstvereenvoudiging voor scholieren in het middelbaar onderwijs met dyslexie te automatiseren. De software bestudeert met welke woorden de gebruiker moeite heeft, en vervangt die door simpelere alternatieven. Hero bevindt zich nu in beta-vorm en wordt enkel in het Engels en Deens ondersteund. Als alternatief is er Readable⁶. Dit is een Engelstalige AI-toepassing dat zinnen beoordeeld met leesbaarheidsformules.

Roldós (2020) haalt aan dat NLP in de laatste decennia volop in ontwikkeling is, maar ontwikkelaars botsen nog op uitdagingen. Het gaat om zowel interpretatie- als dataproblemen bij AI-modellen. Het onderzoek haalt twee punten aan. Allereerst is het voor een machine moeilijk om de context van homoniemen te achterhalen. Bijvoorbeeld bij het woord 'bank' is het niet duidelijk voor de machine of het gaat over de geldinstelling of het meubel. Daarnaast zijn synoniemen een probleem voor tekstverwerking.

Het onderzoek van Sciforce (2020) haalt aan dat het merendeel van NLP-toepassingen Engelstalige invoer gebruikt. Niet-Engelstalige toepassingen zijn zeldzaam. De opkomst van AI technologieën die twee datasets gebruiken, biedt een oplossing voor dit probleem. De software vertaalt eerst de oorspronkelijke tekst naar de gewenste taal, voordat de tekst wordt herwerkt. Hetzelfde onderzoek bewijst dat het vertalen van gelijkaardige talen, zoals Duits en Nederlands, een minimaal verschil opleverd. Volgens Plavén-Sigray e.a. (2017) houden onderzoekers zich vaak in hun eigen taalbubbel, wat negatieve gevolgen heeft voor de leesbaarheid van een wetenschappelijk artikel. Bovendien vormt de stijgende trend van het gebruik aan acroniemen Barnett en Doubleday (2020) een extra hindernis. Donato e.a. (2022) haalt aan dat onbegrijpelijke literatuur, waaronder studiemateriaal geschreven door de docent en online wetenschappelijke artikelen, één van de redenen is waarom scholieren

⁵<https://beta.heroapp.ai/>

⁶<https://readable.com/>

met dyslexie in het middelbaar onderwijs van richting veranderen.

A.2.3. Huidige toepassingen

Vlaanderen heeft weinig zicht op de geïmplementeerde AI software in scholen. Dit werd vastgesteld door (Martens e.a., 2021a), een samenwerking tussen de Vlaamse universiteiten en overheid voor AI. Vergeleken met andere Europese landen, maakt België het minst gebruik van leerling-georiënteerde hulpmiddelen. Degenen die wel gebruikt worden, zijn vooral online leerplatformen voor zelfstandig werken. Ook maakt België amper gebruik van beschikbare software die de leermethoden en -noden van leerlingen evalueert (Martens e.a., 2021b).

Verhoeven (2023) haalt aan dat AI-toepassingen zoals ChatGPT, Google Bard en Bing AI kunnen helpen om routinematig werk te verminderen in het onderwijs. Echter haalt Deckmyn (2021) aan dat GPT-3, het model van ChatGPT, sterker staat in het maken van Engelstalige teksten vergeleken met Nederlandstalige teksten. De databank waar het GPT-3 model mee is getraind, bestaat uit 92% Engelstalige woorden, terwijl er 0,35% Nederlandse woorden aanwezig zijn in dezelfde databank. Ontwikkelaars moeten de evolutie van deze modellen opvolgen, voordat er Nederlandstalige toepassingen mee worden gemaakt.

A.2.4. Ontwikkelen met AI

Python staat bovenaan de lijst van programmeertalen voor NLP-toepassingen. Volgens het onderzoek van Thangarajah (2019) is dit te wijten aan de eenvoudige syntax, kleine leercurve en grote beschikbaarheid van kant-en-klare bibliotheken. Wetenschappelijke berekeningen of statistische analyses kunnen worden uitgevoerd met één lijn code. Malik (2022) haalt de twee meest voorkomende aan, namelijk NLTK⁷ en Spacy⁸. *Deep Martin*⁹ bouwt verder op het onderzoek van Shardlow (2014) naar een pipeline voor lexicale vereenvoudiging. *Deep Martin* maakt gebruik van *custom transformers* om invoertekst te converteren naar een vereenvoudigde versie van de tekstinhoud.

Voor Germaanse talen zijn er enkele datasets en word embeddings beschikbaar die de complexiteit van woorden bijhouden. Zo zijn er in de Duitse taal Klexikon¹⁰ en TextComplexityDE¹¹. Een onderzoek van Suter e.a. (2016) bouwde een rule-based NLP-model met 'Leichte Sprache', wat een dataset is met eenvoudige Duitstalige zinsconstructies. Nederlandstalige datasets zijn in schaarse hoeveelheden beschikbaar, dus het vertalen uit een Germaanse taal is hier een optie.

Volgens Garbacea e.a. (2021) is het belangrijk dat AI-ontwikkelaars niet alleen aandacht besteden aan het aanpassen van woorden en zinnen, maar ook aan de ge-

⁷<https://www.nltk.org/>

⁸<https://spacy.io/>

⁹<https://github.com/chrislemke/deep-martin>

¹⁰<https://github.com/dennlinger/klexikon>

¹¹<https://github.com/babaknaderi/TextComplexityDE>

bruiker meegeven waarom iets is aangepast. De onderzoekers wijzen op twee ethische aspecten. Eerst moet de toepassing duidelijk aangeven waarom een woord of zin is aangepast. Het model moet de moeilijkheidsgraad van de woorden of zinnen bewijzen. Iavarone e.a. (2021) beschrijft een methode met regressiemodellen om de moeilijkheidsgraad te bepalen door een gemiddeld moeilijkheidspercentage per zin te berekenen. Daarnaast benadrukt Garbacea e.a. (2021) het belang van het markeren van de complexere delen van een tekst. Hiervoor haalt hetzelfde onderzoek methoden aan zoals *lexical* of *deep learning*.

Er is een tactvolle aanpak nodig om een vereenvoudigde tekst met AI te beoordelen. De studie van Swayamdipta (2019) haalt aan dat er extra nood is aan NLP-modellen waarbij de tekst zijn kernboodschap behoudt. Samen met Microsoft Research bouwden ze NLP-modellen die gericht waren op de bewaring van zinsstructuur en -context door *scaffolded learning*. Hiervoor maakten de onderzoekers gebruik van een voorspellingsmethode die de positie van woorden en zinnen in een document beoordeelde. De Flesch-Kincaid leesbaarheidstest is volgens Readable (2021) een alternatieve manier om vereenvoudigde tekstinhoud te beoordelen, zonder de nood aan *pre-trained* modellen. Deze score kan eenvoudig worden berekend met de *Python-library textstat*¹².

A.3. Methodologie

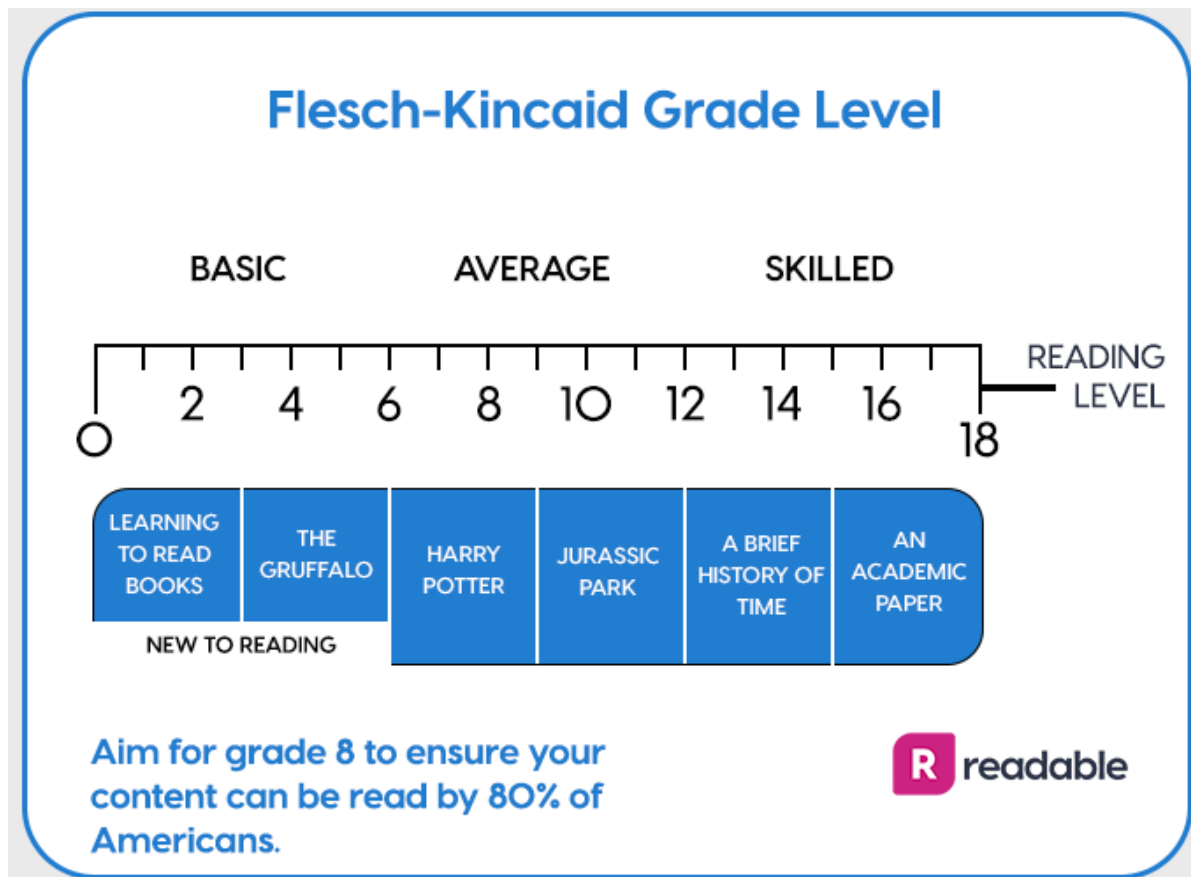
Een *mixed-methods* onderzoek toont aan hoe toepassingen automatisch een wetenschappelijke artikel kunnen vereenvoudigen, gericht op scholieren met dyslexie in de derde graad middelbaar onderwijs. Het onderzoek houdt vijf grote fases in. De eerste fase is het proces van geautomatiseerde tekstvereenvoudiging beschrijven. Dit gebeurt via een grondige studie van vakliteratuur en wetenschappelijke teksten. Ook blogs van experts komen hier aan bod. Na het verwerven van de nodige inzichten wordt er een verklarende tekst opgesteld.

De tweede fase bestaat uit het analyseren van wetenschappelijke werken over de bewezen voordelen van tekstvereenvoudiging bij scholieren met dyslexie van de derde graad middelbaar onderwijs. Hiervoor zijn geringe thesissen beschikbaar, die zorgvuldigheid vragen tijdens interpretatie. De resulterende tekst bevat de voordelen samen met hun wetenschappelijke onderbouwing.

De derde fase is opnieuw een beschrijving. Hier worden de valkuilen bij taalverwerking met AI-software nagegaan. Deze fase van het onderzoek brengt mogelijke nadelen en tekortkomingen van AI-software bij tekstvereenvoudiging aan het licht. Dit gebeurt aan de hand van een technische uitleg.

De vierde fase omvat een toelichting over beschikbare AI toepassingen voor tekstvereenvoudiging. Aan de hand van een veldonderzoek op het internet en bij bedrijven wordt een longlist opgesteld van beschikbare toepassingen voor tekstvereenvoudiging in het middelbaar onderwijs. Met een requirementsanalyse wordt

¹²<https://pypi.org/project/textstat/>



Figuur (A.1)
(Readable, 2021)

er een shortlist opgesteld van software. Het toetsen van verschillende tools wordt ook betrokken in deze fase. De shortlist vormt de basis voor de ontwikkeling van een prototype voor geautomatiseerde en adaptieve tekstvereenvoudiging.

De vijfde en laatste fase van het onderzoek bestaat uit het testen en beoordelen van gekozen AI-toepassingen voor tekstvereenvoudiging. In dit experiment proberen scholieren met dyslexie in de derde graad middelbaar onderwijs de shortlisted AI toepassingen en het prototype uit. Het doel van het experiment is om de effectiviteit en gebruikersvriendelijkheid van deze toepassingen te beoordelen. Na een grondige analyse wordt er met de resultaten bepaalt of de toepassingen aan de unieke noden van een scholier met dyslexie in de derde graad middelbaar onderwijs voldoen om wetenschappelijke artikelen te vereenvoudigen voor scholieren in het middelbaar onderwijs.

A.4. Verwacht resultaat, conclusie

Er wordt verwacht dat de huidige softwareoplossingen voor tekstvereenvoudiging onvoldoende aansluiten bij de noden van scholieren met dyslexie in de derde graad middelbaar onderwijs. Het prototype is moeilijk af te stemmen op de specifieke noden van deze doelgroep. Ontwikkelaars die werken met bestaande modellen moeten *custom transformers* inzetten om bevredigende resultaten te krijgen. Bovendien ontbreken er Nederlandstalige word embeddings die de complexiteit van elk woord bijhouden en aan kant-en-klare modellen die de inhoud van wetenschappelijke artikelen kunnen vereenvoudigen. Word embeddings uit een Germaanse taal gebruiken, gevolgd door vertaling naar het Nederlands is wel een aanvaardbaar alternatief.

B

Code voor tekstanalyse

```
1 import os, re
2
3 def add_newline_after_dot(input_file, output_file):
4     with open(input_file, 'r', encoding='utf-8') as file:
5         text = file.read()
6         text = re.sub(r'\d', '', text)
7         modified_text = text.replace('.', '.\n')
8         with open(output_file, 'w', encoding='utf-8') as file:
9             file.write(modified_text)
10
11 folder_path = 'scripts\pdf'
12 original_scientific_papers = [f for f in os.listdir(folder_path)]
13
14 for paper in original_scientific_papers:
15     input_file = folder_path + '/' + paper
16     output_file = folder_path + '/' + 'RE_' + paper
17     add_newline_after_dot(input_file, output_file)
```

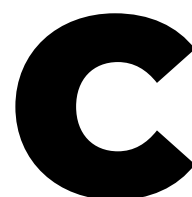
Listing B.1: Script voor fase 1 van de vergelijkende studie.

```
1 import os
2 import pandas as pd
3 from deep_translator import GoogleTranslator
4
5 output_csv = 'results.csv'
6
7 def translate_dutch_to_english(dutch_text_file):
8     with open(dutch_text_file, 'r', encoding='utf-8') as file:
9         dutch_sentences = file.readlines()
10        dutch_sentences = [sentence.strip() for sentence in dutch_sentences]
11
12        english_sentences = []
13        for sentence in dutch_sentences:
14            translated = GoogleTranslator(source='nl', target='en').translate(sentence)
15            english_sentences.append(translated)
16        df = pd.DataFrame({'Dutch': dutch_sentences, 'English': english_sentences})
17        df.to_csv(str(dutch_text_file).split('.')[0] + '.csv', index=False)
18
19
20 folder_path = 'scripts/pdf/'
21 original_scientific_papers = [f for f in os.listdir(folder_path)]
22
23 for paper in original_scientific_papers:
24     if paper.startswith('RE_') and paper.endswith('.txt'):
25         print(f'STARTING {paper}')
26         dutch_text_file = folder_path + paper
27         translate_dutch_to_english(dutch_text_file)
28
```

Listing B.2: Script voor de tweede fase van de vergelijkende studie.

```
1 import os
2 from langdetect import detect
3 import spacy
4 import pandas as pd
5 import readability
6
7 simplified_folder = 'scripts/vereenvoudigde_artikelen'
8 original_folder = 'scripts/pdf'
9
10 scientific_papers = [original_folder + "/" + f for f in os.listdir(original_folder)
11                       ] + [simplified_folder + "/" + f for f in os.listdir(simplified_folder)]
12
13 languages = {
14     'nl': 'nl_core_news_md',
15     'en': 'en_core_web_md'
16 }
17 df = pd.DataFrame()
18
19 for paper in scientific_papers:
20     with open(paper, 'r', encoding='utf-8') as file:
21         text = file.read()
22         nlp = spacy.load(languages.get('nl'))
23         doc = nlp(text)
24
25         for sent in doc.sents:
26             try:
27                 metrics = readability.getmeasures(sent.text, lang='nl')
28                 row = {
29                     'Paper': paper.split('/')[2].split('.')[0],
30                     'Sentence': sent.text,
31                     'FRE': metrics['readability grades']['FleschReadingEase'],
32                     'FOG': metrics['readability grades']['GunningFogIndex'],
33                 }
34
35                 for key, value in metrics['sentence info'].items():
36                     row[key] = value
37
38                 for key, value in metrics['word usage'].items():
39                     row[key] = value
40
41                 for key, value in metrics['sentence beginnings'].items():
42                     row[key] = value
43
44                 df = df.append(row, ignore_index=True)
45             except Exception as e:
46                 print(e)
47
48 df.to_csv('result.csv', index=False)
49
```

Listing B.3: Script voor fase 4 van de vergelijkende studie



Code voor het prototype

```
1 SENTENCES\_PER\_PARAGRAPH = 5
2
3 dict = {
4     'nl': 'nl_core_news_md',
5     'en': 'en_core_web_md'
6 }
7
8
9 def get_full_text_dict(self, all_pages):
10     total = ""
11     for page_layout in all_pages:
12         for element in page_layout:
13             if isinstance(element, LTTextContainer):
14                 for text_line in element:
15                     total += text_line.get_text()
16     return total
17
18
19 def get_full_text_from_image(self, all_pages):
20     img_files = []
21     num_pages = 0
22     for i, page in enumerate(all_pages):
23         file = f'page_{num_pages}.jpg'
24         page.save(file, 'JPEG')
25         img_files.append(file)
26         num_pages += 1
27
28     full_text = []
29     reader = easyocr.Reader(['nl'])
30     for f in img_files:
31         result = reader.readtext(f, detail=0)
32         full_text.append(" ".join(result))
33     os.remove(f)
```

```

34
35     return " ".join(full_text)
36
37 def get_full_text_site(self, full_text):
38     try:
39         lang = detect(full_text)
40     except:
41         lang = 'en'
42
43     if lang in dict:
44         nlp = spacy.load(dict.get(lang))
45     else:
46         nlp = spacy.load(dict.get('en'))
47
48     full_text = str(full_text).replace('\n', ' ')
49
50     doc = nlp(full_text)
51     sentences = []
52     for sentence in doc.sents:
53         sentences.append(sentence)
54
55     pad_size = SENTENCES_PER_PARAGRAPH - (len(sentences) % SENTENCES_PER_PARAGRAPH)
56
57     padded_a = np.pad(sentences, (0, pad_size), mode='empty')
58     paragraphs = padded_a.reshape(-1, SENTENCES_PER_PARAGRAPH)
59
60     text_w_pos = []
61     for paragraph in paragraphs:
62         paragraph_w_pos = []
63         try:
64             for sentence in paragraph:
65                 dict_sentence = {}
66                 for token in sentence:
67                     dict_sentence[token.text] = str(token.pos_).lower()
68                 paragraph_w_pos.append(dict_sentence)
69             text_w_pos.append(paragraph_w_pos)
70         except:
71             pass
72
73     return text_w_pos

```

Listing C.1: Reader-klasse

```

1 class HuggingFaceModels:
2     def __init__(self, key=None):
3         global huggingface_api_key
4         try:
5             huggingface_api_key = key
6         except:
7             huggingface_api_key = 'not_submitted'
8

```



```

9  """
10 def query(self, payload, API_URL):
11     headers = {"Authorization": f"Bearer {huggingface_api_key}"}
12     response = requests.post(API_URL, headers=headers, json=payload)
13     return response.json()
14
15 """
16 def scientific_simplify(self, text, lm_key):
17     length = len(text)
18     API_URL = huggingfacemodels.get(lm_key)
19     gt = Translator()
20     translated_text = gt.translate(text=text, src='nl', dest='en').text
21     result = self.query({"inputs": str(translated_text), "parameters": {"max_length": length}, "options": {"wait_for_model": True}}, API_URL)[0]['generated_text']
22     result = gt.translate(text=result, src='en', dest='nl').text
23     return result
24
25 def summarize(self, text, lm_key):
26     gt = Translator()
27     soup = BeautifulSoup(text, 'html.parser')
28     tags = soup.find_all(True)
29     split_text = {}
30     for tag in tags:
31         if tag.name == 'h3':
32             current_key = tag.text
33         if tag.name == 'p':
34             split_text[current_key] = tag.text
35
36     for key in split_text.keys():
37         split_text[key] = str(split_text[key]).strip('\n').replace('\n', ' ').replace('\ ', ' ')
38
39     result_dict = {}
40     for key in split_text.keys():
41         text = split_text[key]
42         origin_lang = detect(text)
43         nlp = spacy.load(languages.get(origin_lang, 'en'))
44         doc = nlp(text)
45
46     sentences = []
47     for s in doc.sents:
48         try:
49             text = gt.translate(text=str(s), dest='en').text
50             sentences.append(text)
51         except Exception as e:
52             print(e)
53
54     API_URL = huggingfacemodels.get(lm_key)
55     sentences = np.array(sentences)
56     pad_size = 3 - (sentences.size % 3)
57     padded_a = np.pad(sentences, (0, pad_size), mode='empty')

```

```

58     paragraphs = padded_a.reshape(-1, 3)
59
60     output = []
61     text = ""
62     for i in paragraphs:
63         length = len(str(i))
64         result = self.query({"inputs": str(i), "parameters": {"max_length": length}, "
options":{"wait_for_model":True}}, API_URL)
65
66     try:
67         if 'generated_text' in result[0]:
68             text = result[0].get('generated_text')
69
70         if 'summary_text' in result[0]:
71             text = result[0].get('summary_text')
72     except Exception as e:
73         print(e)
74
75     lang = detect(text)
76     try:
77         text = gt.translate(text=str(text), src=lang, dest='nl').text
78     except Exception as e:
79         print(str(e))
80
81     output.append(text)
82     result_dict[key] = output
83     return(result_dict)
84
85     """@returns a translated sentence"""
86     def translate_sentence(self, sentence):
87         translator = Translator()
88         result = translator.translate(
89             text=sentence,
90             dest='nl'
91         )
92         return result.text

```

Listing C.2: HuggingFace-klasse

```

1 class GPT():
2     """ @sets openai.api_key """
3     def __init__(self, key=None):
4         global gpt_api_key
5         if key is None:
6             gpt_api_key = 'not-submitted'
7             openai.api_key = key
8         else:
9             gpt_api_key = key
10            openai.api_key = key
11
12    """ @returns prompt, result from gpt """

```

```

13 def look_up_word_gpt(self, word, context):
14     try:
15         prompt = f"""
16         Give a simple Dutch explanation in one sentence for this word in the given
17         context. Give the PoS-tag and Dutch definition: '{word}'
18         context: {context}
19         format: PoS-tag | definition
20         ///
21         """
22         result = openai.Completion.create(
23             prompt=prompt,
24             temperature=0,
25             max_tokens=50,
26             model=COMPLETIONS_MODEL,
27             top_p=0.9,
28             stream=False
29         )["choices"][0]["text"].strip(" \n")
30         return result, word, prompt
31     except Exception as e:
32         return 'error', str(e), str(e)
33
34     """ @returns prompt, result from gpt """
35 def give_synonym(self, word, context):
36     try:
37         prompt = f"""
38         Give a Dutch synonym for '{word}'. If there is no Dutch synonym available,
39         explain it between curly brackets.
40         context:
41         {context}
42         """
43         result = openai.Completion.create(
44             prompt=prompt,
45             temperature=0,
46             max_tokens=10,
47             model=COMPLETIONS_MODEL,
48             top_p=0.9,
49             stream=False
50         )["choices"][0]["text"].strip(" \n")
51         return result, word, prompt
52     except Exception as e:
53         return 'Open AI outage of problemen', str(e)
54
55 def personalised_simplify(self, sentence, personalisation):
56     if 'summary' in personalisation:
57         prompt = f"""
58         Simplify the sentences in the given text and {"", ".join(personalisation)}
59         :return: A list of simplified sentences divided by a '|' sign
60         ///
61         {sentence}

```

```

62     """
63     else:
64         prompt = f"""
65         Explain this in own Dutch words and {"", ".join(personalisation)}
66         ///
67         {sentence}
68         """
69
70     try:
71         result = openai.Completion.create(
72             prompt=prompt,
73             temperature=0,
74             max_tokens=len(prompt) ,
75             model=COMPLETIONS_MODEL,
76             top_p=0.9,
77             stream=False
78         )["choices"][0]["text"].strip(" \n")
79
80         if 'summary' in personalisation:
81             result = result.split('|')
82         else:
83             result = [result]
84
85         return result , prompt
86     except Exception as e:
87         return str(e) , prompt
88
89     def personalised_simplify_w_prompt(self , sentences , personalisation):
90         try:
91             result = openai.Completion.create(
92                 prompt=personalisation ,
93                 temperature=0,
94                 max_tokens=len(personalisation)+len(sentences) ,
95                 model=COMPLETIONS_MODEL,
96                 top_p=0.9,
97                 stream=False
98             )["choices"][0]["text"].strip(" \n")
99             return result , personalisation
100         except Exception as e:
101             return str(e) , personalisation
102
103
104     def summarize(self , full_text_dict , personalisation):
105         soup = BeautifulSoup(full_text_dict , 'html.parser')
106         tags = soup.find_all(True)
107         split_text = {}
108
109         for tag in tags:
110             if tag.name == 'h3':
111                 current_key = tag.text
112 
```

```

113     if tag.name == 'p':
114         split_text[current_key] = tag.text
115
116     for key in split_text.keys():
117         split_text[key] = str(split_text[key]).strip('\n')\
118             .strip('\\').replace('\\', '')
119
120     new_text = {}
121     for title in split_text.keys():
122         text = split_text[title]
123         if len(text) > 1000:
124             index = len(text) // 2
125             text_to_prompt = [text[:index], text[index:]]
126         else:
127             text_to_prompt = [text]
128
129     full_chunk_result = ""
130     for chunk in text_to_prompt:
131         # TODO aangepast voor oorspronkelijke taal
132         if 'summation' not in personalisation:
133             prompt = f"""
134             Rewrite this with {"", ".join(personalisation)}
135             ///
136             {chunk}
137             """
138         else:
139             prompt = f"""
140             Rewrite this as a list of simplified Dutch sentences with {"", ".join(
141             personalisation)}
142             :return: A list of simplified sentences divided by a '|' sign
143             ///
144             {chunk}
145             """
146
147         full_chunk_result += str(openai.Completion.create(prompt=prompt, temperature
148         =0, max_tokens=500, model=COMPLETIONS_MODEL, top_p=0.9, stream=False) ["choices"
149         ] [0] ["text"].strip(" \n"))
150     new_text[title] = [full_chunk_result]
151     return new_text

```

Listing C.3: Reader-klasse

```

1 import subprocess, io, os, py pandoc
2 from datetime import date
3 import zipfile
4
5
6 markdown_file = "saved_files/file.md"
7 zip_filename = 'saved_files/simplified_docs.zip'
8 pdf_file = "saved_files/output.pdf"

```

```

9 docx_file = "saved_files/output.docx"
10 DATE_NOW = str(date.today())
11
12
13 class Creator():
14     def create_header(self, title, margin, fontsize, chosen_font, chosen_title_font,
15         word_spacing, type_spacing):
16         with open(markdown_file, 'w', encoding='utf-8') as f:
17             f.write("---\n")
18             f.write(f"title: {title}\n")
19             f.write(f"mainfont: {chosen_font}.ttf\n")
20             f.write(f"titlefont: {chosen_title_font}.ttf\n")
21             f.write(f"date: {DATE_NOW}\n")
22             f.write(f"document: article\n")
23             f.write(f"geometry: margin={margin}cm\n")
24             f.write(f"fontsize: {fontsize}pt\n")
25             f.write('header-includes:\n')
26             f.write(f'- \spaceskip={word_spacing}cm\n')
27             f.write(f'- \usepackage{{setspace}}\n')
28             f.write(f'- \{type_spacing}\n')
29             f.write("---\n")
30
31     def generate_glossary(self, list):
32         with open(markdown_file, 'a', encoding='utf-8') as f:
33             f.write("---\n")
34             f.write("# Woordenlijst\n")
35             f.write("| Woord | Soort | Definitie |\n")
36             f.write("| --- | --- | --- |\n")
37             for word in list.keys():
38                 f.write(f"| {word} | {list[word]['type']} | {list[word]['definition']} |\n")
39             f.write("\n")
40
41     def generate_summary(self, full_text):
42         with open(markdown_file, 'a', encoding="utf-8", errors="surrogateescape") as f:
43             for key in full_text.keys():
44                 title = str(key).replace('\n', ' ')
45                 text = full_text[key]
46                 f.write('\n\n')
47                 f.write(f'## {title}')
48                 f.write('\n\n')
49                 f.write(" ".join(text))
50                 f.write('\n\n')
51
52
53     def generate_summary_w_summation(self, full_text):
54         with open(markdown_file, 'a', encoding="utf-8", errors="surrogateescape") as f:
55             for key in full_text.keys():
56                 title = str(key).replace('\n', ' ')
57                 text = full_text[key][0].split('|')

```

```

58         f.write('\n\n')
59         f.write(f'## {title}')
60         for sentence in text:
61             f.write('\n\n')
62             f.write(f'* {sentence}')
63             f.write('\n\n')
64
65
66     def create_pdf(self, title, margin, list, full_text, fonts, word_spacing,
67                   type_spacing, summation):
68         if title is not None:
69             self.create_header(title=title, margin=margin, fontsize=14, chosen_font=
70                               fonts[0], chosen_title_font=fonts[1], word_spacing=word_spacing, type_spacing=
71                               type_spacing)
69         else:
70             self.create_header(title='Vereenvoudigde tekst', margin=0.5, fontsize=14,
71                               chosen_font=fonts[0], chosen_title_font=fonts[1], word_spacing=word_spacing,
72                               type_spacing=type_spacing)
71
72         """GLOSSARY"""
73         if len(list) != 0:
74             self.generate_glossary(list=list)
75
76         """SUMMARY"""
77         if summation:
78             self.generate_summary_w_summation(full_text=full_text)
79         else:
80             self.generate_summary(full_text=full_text)
81
82         """FILE_CREATION"""
83         py pandoc.convert_file(source_file=markdown_file, to='docx', outputfile=
84                               docx_file, extra_args=["-M2GB", "+RTS", "-K64m", "-RTS"])
84         py pandoc.convert_file(source_file=markdown_file, to='pdf', outputfile=
85                               pdf_file, extra_args=['--pdf-engine=xelatex'])
85         with zipfile.ZipFile(zip_filename, 'w') as myzip:
86             myzip.write(pdf_file)
87             myzip.write(docx_file)

```

Listing C.4: Writer-klasse omvattende de code om dynamische PDF- en Word-documenten te genereren.

```

1 @echo off
2
3 cd web-app
4 docker stop text-application-prototype
5 docker rm text-application-prototype
6
7 docker rmi text-app
8
9 docker build -t text-app .
10 docker run --name text-application-prototype --network webapp_simplification -d -p
    5000:5000 text-app

```

Listing C.5: Script voor het opstarten van de Docker-container voor Windows-gebruikers

```
1 #!/bin/sh
2
3 cd web-app || exit
4 docker stop text-application-prototype
5 docker rm text-application-prototype
6
7 docker rmi text-app
8
9 docker build -t text-app .
10 docker run --name text-application-prototype --network webapp_simplification -d -p
    5000:5000 text-app
```

Listing C.6: Script voor het opstarten van de Docker-container voor Unix-gebruikers

Bibliografie

- (2020). <https://www.vlaanderen.be/publicaties/begrijpend-leesonderwijs-in-de-basisscholen-kwaliteitsvol-sterke-en-zwakke-punten-van-de-huidige-praktijk>
- (2021). <https://www.cwauthors.com/article/How-to-write-about-complex-scientific-concepts-in-simple-accessible-language>
- (2023). <https://accessibility.huit.harvard.edu/disabilities/dyslexia>
- Althunayyan, S. & Azmi, A. (2021). Automated Text Simplification: A Survey. *ACM Computing Surveys*, 54, Article no. 43. <https://doi.org/10.1145/3442695>
- Ball, P. (2017). It's not just you: science papers are getting harder to read. *Nature*.
- Barnett, A. & Doubleday, Z. (2020). Meta-Research: The growth of acronyms in the scientific literature (P. Rodgers, Red.). *eLife*, 9, e60080.
- Belpaeme, T., Kennedy, J., Ramachandran, A., Scassellati, B. & Tanaka, F. (2018). Social robots for education: A review. *Science robotics*, 3(21), eaat5954.
- Bezem, A. & Lugthart, M. (2016). Visuele Disfunctie een onzichtbare belemmering bij lezen, spelling en concentratie. <https://beeldenbrein.nl/>
- Bilici, Ş. (2021). Sequence labeling.
- Bingel, J., Paetzold, G. & Søgaaard, A. (2018). Lexi: A tool for adaptive, personalized text simplification. *Proceedings of the 27th International Conference on Computational Linguistics*, 245–258.
- Binz, M. & Schulz, E. (2023). Using cognitive psychology to understand GPT-3. *Proceedings of the National Academy of Sciences*, 120(6).
- Bonte, M. (2020). *Bestaat Dyslexie?: En is het een relevante vraag?* uitgeverij SWP.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language Models are Few-Shot Learners.
- Bulté, B., Sevens, L. & Vandeghinste, V. (2018). Automating lexical simplification in Dutch. *Computational Linguistics in the Netherlands Journal*, 8, 24–48. <https://clinjournal.org/clinj/article/view/78>
- Canning, Y., Tait, J., Archibald, J. & Crawley, R. (2000). Cohesive Generation of Syntactically Simplified Newspaper Text. In P. Sojka, I. Kopeček & K. Pala (Red.), *Text, Speech and Dialogue* (pp. 145–150). Springer Berlin Heidelberg.
- Cao, M. (2022). A Survey on Neural Abstractive Summarization Methods and Factual Consistency of Summarization.

- Chowdhary, K. (2020). *Fundamentals of Artificial Intelligence*. Springer, New Delhi.
- Coster, W. & Kauchak, D. (2011). Learning to Simplify Sentences Using Wikipedia. *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, 1–9. <https://aclanthology.org/W11-1601>
- Crevits, H. (2022, maart 13). *Kwart van bedrijven gebruikt artificiële intelligentie: Vlaanderen bij beste leerlingen van de klas* (Persbericht). Vlaamse Overheid Departement Economie, Wetenschap en Innovatie.
- Dandekar, N. (2016). How to use machine learning to find synonyms. <https://medium.com/@nikhilbd/how-to-use-machine-learning-to-find-synonyms-6380c0c6106b>
- Dapaah, J. & Maenhout, K. (2022, juli 8). *Iedereen heeft boter op zijn hoofd* (D. Standaard, Red.). https://www.standaard.be/cnt/dmf20220607_97763592
- De Craemer, J., Van Beeumen, L., Cooreman, A., Moonen, A., Rottier, J., Wagemakers, I. & Mardulier, T. (2018). Aan de slag met voorleessoftware op school. Een gids met 8 vragen en antwoorden. <https://onderwijs.vlaanderen.be/nl/onderwijspersoneel/van-basis-tot-volwassenenonderwijs/lespraktijk/ict-in-de-klas/voorleessoftware-voor-leerlingen-met-leesbeperkingen/aan-de-slag-met-voorleessoftware-op-school>
- De Meyer, I., Janssens, R. & Warlop, N. (2019). Leesvaardigheid van 15- jarigen in Vlaanderen: Overzicht van de eerste resultaten van PISA2018. <https://data-onderwijs.vlaanderen.be/documenten/bestand.ashx?id=12265>
- Deckmyn, D. (2021, maart 19). *Robot schrijft mee De Standaard* (D. Standaard, Red.). https://www.standaard.be/cnt/dmf20210319_05008561
- Donato, A., Muscolo, M., Arias Romero, M., Capri, T., Calarese, T. & Olmedo Moreno, E. M. (2022). Students with dyslexia between school and university: Post-diploma choices and the reasons that determine them. An Italian study. *Dyslexia*, 28(1), 110–127.
- DuBay, W. H. (2004). The principles of readability. *Online Submission*.
- Eisenstein, J. (2019). *Introduction to Natural Language Processing*. MIT Press. <https://books.google.be/books?id=72yuDwAAQBAJ>
- Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R. & Radev, D. (2020). SummEval: Re-evaluating Summarization Evaluation.
- Gala, N. & Ziegler, J. (2016). Reducing lexical complexity as a tool to increase text accessibility for children with dyslexia. *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)*, 59–66.
- Garbacea, C., Guo, M., Carton, S. & Mei, Q. (2021). Explainable Prediction of Text Complexity: The Missing Preliminaries for Text Simplification. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1086–1097. <https://doi.org/10.18653/v1/2021.acl-long.88>

- Garg, H. (2022). Using GPT-3 for education: Use cases. <https://indiaai.gov.in/article/using-gpt-3-for-education-use-cases>
- Ghesquière, P. (2018). *Als leren pijn doet: Kinderen met een leerstoornis opvoeden en begeleiden*. Acco.
- Gooding, S. (2022). On the Ethical Considerations of Text Simplification. *Ninth Workshop on Speech and Language Processing for Assistive Technologies (SLPAT-2022)*, 50–57. <https://doi.org/10.18653/v1/2022.slpac-1.7>
- Gooding, S. & Kochmar, E. (2019). Complex word identification as a sequence labelling task. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1148–1153.
- Greg, B., Atty, E., Elie, G., Joane, J., Logan, K., Lim, R., Luke, M. & Michelle, P. (2023). Introducing chatgpt and Whisper Apis. <https://openai.com/blog/introducing-chatgpt-and-whisper-apis>
- Hahn, U. & Mani, I. (2000). The Challenges of Automatic Summarization. *Computer*, 33, 29–36. <https://doi.org/10.1109/2.881692>
- Hartley, J. (1999). From Structured Abstracts to Structured Articles: A Modest Proposal. *Journal of Technical Writing and Communication*, 29(3), 255–270. <https://doi.org/10.2190/3RWW-A579-HC8W-6866>
- Harwell, D. (2023). Tech's hottest new job: Ai whisperer. no coding required. <https://www.washingtonpost.com/technology/2023/02/25/prompt-engineers-techs-next-big-job/>
- Hayes, D. P. (1992). The growing inaccessibility of science. <https://www.nature.com/articles/356739a0>
- Hern, A. (2023). TechScape: Will meta's massive leak democratise AI – and at what cost? <https://www.theguardian.com/technology/2023/mar/07/techscape-meta-leak-llama-chatgpt-ai-crossroads>
- Hollenkamp, J. (2020). Summary and analysis of Scientific Research Articles - San Jose State ... <https://www.sjsu.edu/writingcenter/docs/handouts/Summary%20and%20Analysis%20of%20Scientific%20Research%20Articles.pdf>
- Hsu, W.-T., Lin, C.-K., Lee, M.-Y., Min, K., Tang, J. & Sun, M. (2018). A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss.
- Huang, S., Wang, R., Xie, Q., Li, L. & Liu, Y. (2019). An Extraction-Abstraction Hybrid Approach for Long Document Summarization. *2019 6th International Conference on Behavioral, Economic and Socio-Cultural Computing (BESCC)*, 1–6.
- Hubbard, K. E. & Dunbar, S. D. (2017). Perceptions of scientific research literature and strategies for reading papers depend on academic career stage. *PLOS ONE*, 12(12), 1–16.

- Iavarone, B., Brunato, D. & Dell'Orletta, F. (2021). Sentence Complexity in Context. *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, 186–199. <https://doi.org/10.18653/v1/2021.cmcl-1.23>
- IBM. (2022). IBM Global AI Adoption Index 2022. <https://www.ibm.com/downloads/cas/GVAGA3JP>
- Iredale, G. (2022). An overview of tokenization algorithms in NLP. <https://101blockchains.com/tokenization-nlp/>
- Iskender, N., Polzehl, T. & Möller, S. (2021). Reliability of Human Evaluation for Text Summarization: Lessons Learned and Challenges Ahead. *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, 86–96. <https://aclanthology.org/2021.humeval-1.10>
- Jiang, R. K. (2023). Prompt engineering : Deconstructing and managing intention. <https://www.linkedin.com/pulse/prompt-engineering-deconstructing-managing-intention-jiang/>
- Jones, R., Colusso, L., Reinecke, K. & Hsieh, G. (2019). r/science: Challenges and Opportunities in Online Science Communication. *CHI '19: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–14. <https://doi.org/10.1145/3290605.3300383>
- Jurafsky, D., Martin, J., Norvig, P. & Russell, S. (2014). *Speech and Language Processing*. Pearson Education. <https://books.google.be/books?id=Cq2gBwAAQBAJ>
- Kandula, S., Curtis, D. & Zeng-Treitler, Q. (2010). A semantic and syntactic text simplification tool for health content. *AMIA annual symposium proceedings, 2010*, 366.
- Khan, A. (2014). A Review on Abstractive Summarization Methods. *Journal of Theoretical and Applied Information Technology*, 59, 64–72.
- Khurana, D., Koli, A., Khatter, K. & Singh, S. (2022). Natural Language Processing: State of The Art, Current Trends and Challenges. *Multimedia Tools and Applications*, 82, 25–27.
- Kraft, M. A. (2020). Interpreting Effect Sizes of Education Interventions. *Educational Researcher*, 49(4), 241–253. <https://doi.org/10.3102/0013189X20912798>
- Li, C. (2022). OpenAI's GPT-3 language model: A technical overview. <https://lambdalabs.com/blog/demystifying-gpt-3>
- Li, J., Sun, A., Han, J. & Li, C. (2018). A Survey on Deep Learning for Named Entity Recognition.
- Lin, H. & Bilmes, J. (2010). Multi-document summarization via budgeted maximization of submodular functions. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 912–920.
- Linderholm, T., Everson, M. G., van den Broek, P., Mischinski, M., Crittenden, A. & Samuels, J. (2000). Effects of Causal Text Revisions on More- and Less-Skilled

- Readers' Comprehension of Easy and Difficult Texts. *Cognition and Instruction*, 18(4), 525–556.
- Lissens, F., Asmar, M., Willems, D., Van Damme, J., De Coster, S., Demeestere, E., Maes, R., Baccarne, B., Robaeyst, B., Duthoo, W. & Desoete, A. (2020). Het stopt nooit...De impact van dyslexie en/of dyscalculie op het welbevinden en studeren van (jong)volwassenen en op de transitie naar de arbeidsmarkt: een bundeling van Vlaamse pilootstudies.
- Liu, Q., Kusner, M. J. & Blunsom, P. (2020). A Survey on Contextual Embeddings.
- Malik, R. S. (2022, juli 4). *Top 5 NLP Libraries To Use in Your Projects* (T. Al, Red.). <https://towardsai.net/p/l/top-5-nlp-libraries-to-use-in-your-projects>
- Martens, M., De Wolf, R. & Evens, T. (2021a). *Algoritmes en AI in de onderwijscontext: Een studie naar de perceptie, mening en houding van leerlingen en ouders in Vlaanderen*. Kenniscentrum Data en Maatschappij. Verkregen 30 maart 2022, van <https://data-en-maatschappij.ai/publicaties/survey-onderwijs-2021>
- Martens, M., De Wolf, R. & Evens, T. (2021b, juni 28). *School innovation forum 2021*. Kenniscentrum Data en Maatschappij. Verkregen 1 april 2022, van <https://data-en-maatschappij.ai/nieuws/school-innovation-forum-2021>
- McCombes, S. (2022). How to write A summary: Guide amp; examples. <https://www.scribbr.com/working-with-sources/how-to-summarize/>
- McDonald, R. (2007). A study of global inference algorithms in multi-document summarization. *Advances in Information Retrieval: 29th European Conference on IR Research, ECIR 2007, Rome, Italy, April 2-5, 2007. Proceedings* 29, 557–564.
- McFarland, A. (2023). What is prompt engineering in AI amp; Why It Matters. <https://www.unite.ai/what-is-prompt-engineering-in-ai-why-it-matters/>
- McKeown, K., Klavans, J. L., Hatzivassiloglou, V., Barzilay, R. & Eskin, E. (1999). Towards multidocument summarization by reformulation: Progress and prospects.
- McNutt, M. (2014). Reproducibility. *Science*, 343(6168), 229–229. <https://doi.org/10.1126/science.1250475>
- Menzli, A. (2023). Tokenization in NLP: Types, challenges, examples, tools. <https://neptune.ai/blog/tokenization-in-nlp>
- Miszczak, P. (2023). Prompt engineering: The ultimate guide 2023 [GPT-3 amp; chatgpt]. <https://businessolution.org/prompt-engineering/>
- Mottes, C. (2023). GPT-3 vs. Bert: Comparing the two most popular language models. <https://blog.invgate.com/gpt-3-vs-bert>
- Nallapati, R., Zhai, F. & Zhou, B. (2017). SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1). <https://doi.org/10.1609/aaai.v31i1.10958>

- Nandhini, K. & Balasundaram, S. (2013). Improving readability through extractive summarization for learners with reading difficulties. *Egyptian Informatics Journal*, 14(3), 195–204.
- Nenkova, A. & Passonneau, R. (2004). Evaluating Content Selection in Summarization: The Pyramid Method. *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, 145–152.
- Niemeijer, A., Frederiks, B., Riphagen, I., Legemaate, J., Eefsting, J. & Hertogh, C. (2010). Ethical and practical concerns of surveillance technologies in residential care for people with dementia or intellectual disabilities: an overview of the literature. *Psychogeriatrics*, 22(7), 1129–1142. <https://doi.org/10.1017/S1041610210000037>
- OnderwijsVlaanderen. (2023). Voorleessoftware voor Leerlingen met Leesbeperkingen. <https://onderwijs.vlaanderen.be/voorleessoftware-voor-leerlingen-met-leesbeperkingen>
- Paetzold, G. & Specia, L. (2016). SemEval 2016 Task 11: Complex Word Identification. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 560–569. <https://doi.org/10.18653/v1/S16-1085>
- Pain, E. (2016). How to (seriously) read a scientific paper. <https://www.science.org/content/article/how-seriously-read-scientific-paper>
- Plavén-Sigra, P., Matheson, G. J., Schiffler, B. C. & Thompson, W. H. (2017). Research: The readability of scientific texts is decreasing over time (S. King, Red.). *eLife*, 6, e27725.
- Poel, M., Boschman, E. & op den Akker, R. (2008). A Neural Network Based Dutch Part of Speech Tagger [http://eprints.ewi.utwente.nl/14662; 20th Benelux Conference on Artificial Intelligence, BNAIC 2008, BNAIC ; Conference date: 30-10-2008 Through 31-10-2008]. In A. Nijholt, M. Pantic, M. Poel & H. Hondorp (Red.), *BNAIC 2008* (pp. 217–224). Twente University Press (TUP).
- Premjith, P., John, A. & Wilscy, M. (2015). Metaheuristic Optimization Using Sentence Level Semantics for Extractive Document Summarization, 347–358. https://doi.org/10.1007/978-3-319-26832-3_33
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I. e.a. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8), 9.
- Rani, R. & Kaur, B. (2021). The TEXT SUMMARIZATION AND ITS EVALUATION TECHNIQUE. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(1), 745–752.
- Readable. (2021). *Flesch Reading Ease and the Flesch Kincaid Grade Level*. <https://readable.com/readability/flesch-reading-ease-flesch-kincaid-grade-level/>

- Rello, L. & Baeza-Yates, R. (2013). Good fonts for dyslexia. *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2013*.
- Rello, L., Baeza-Yates, R., Dempere-Marco, L. & Saggion, H. (2013). Frequent Words Improve Readability and Short Words Improve Understandability for People with Dyslexia.
- Rello, L., Baeza-Yates, R. & Saggion, H. (2013). The Impact of Lexical Simplification by Verbal Paraphrases for People with and without Dyslexia. *7817*, 501–512.
- Rello, L. & Baeza-Yates, R. A. (2015). How to present more readable text for people with dyslexia. *Universal Access in the Information Society*, *16*, 29–49.
- Rello, L. & Bigham, J. (2017). Good Background Colors for Readers: A Study of People with and without Dyslexia, 72–80.
- Rello, L., Kanvinde, G. & Baeza-Yates, R. (2012). Layout Guidelines for Web Text and a Web Service to Improve Accessibility for Dyslexics. *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility*.
- Ribas, J. (2023). Building the new bing. <https://www.linkedin.com/pulse/building-new-bing-jordi-ribas/>
- Ribeiro, E., Ribeiro, R. & de Matos, D. M. (2018). A Study on Dialog Act Recognition using Character-Level Tokenization.
- Rijkhoff, J. (2022). Tekst Inkorten?: 9 tips om Je Teksten korter Te Maken. <https://dialogtrainers.nl/tekst-inkorten-tips/>
- Rivero-Contreras, M., Engelhardt, P. E. & Saldaña, D. (2021). An experimental eye-tracking study of text adaptation for readers with dyslexia: effects of visual support and word frequency. *Annals of Dyslexia*, *71*, 170–187.
- Roldós, I. (2020, december 22). *Major Challenges of Natural Language Processing (NLP)*. MonkeyLearn. Verkregen 1 april 2022, van <https://monkeylearn.com/blog/natural-language-processing-challenges/>
- Roose, K. (2023). Don't ban chatgpt in schools. teach with it. <https://www.nytimes.com/2023/01/12/technology/chatgpt-schools-teachers.html>
- Santana, V., Oliveira, R., Almeida, L. & Baranauskas, M. C. (2012). Web accessibility and people with dyslexia: A survey on techniques and guidelines. *W4A 2012 - International Cross-Disciplinary Conference on Web Accessibility*. <https://doi.org/10.1145/2207016.2207047>
- Sciforce. (2020, februari 4). *Biggest Open Problems in Natural Language Processing*. Verkregen 1 april 2022, van <https://medium.com/sciforce/biggest-open-problems-in-natural-language-processing-7eb101ccfc9>
- Shardlow, M. (2013). A Comparison of Techniques to Automatically Identify Complex Words. *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*, 103–109. <https://aclanthology.org/P13-3015>

- Shardlow, M. (2014). A Survey of Automated Text Simplification. *International Journal of Advanced Computer Science and Applications(IJACSA), Special Issue on Natural Language Processing 2014*, 4(1). <https://doi.org/10.14569/SpecialIssue.2014.040109>
- Siddharthan, A. (2006). Syntactic Simplification and Text Cohesion. *Research on Language and Computation*, 4(1), 77–109. <http://oro.open.ac.uk/58888/>
- Siddharthan, A. (2014). A survey of research on text simplification. *ITL - International Journal of Applied Linguistics*, 165, 259–298.
- Sikka, P. & Mago, V. (2020). A Survey on Text Simplification. *CoRR*, abs/2008.08612. <https://arxiv.org/abs/2008.08612>
- Simon, J. (2021). Large language models: A new moore's law? <https://huggingface.co/blog/large-language-models>
- Snow, C. (2010). Academic Language and the Challenge of Reading for Learning About Science. *Science (New York, N.Y.)*, 328, 450–2.
- Sohom, G., Ghosh; Dwight. (2019). *Natural Language Processing Fundamentals*. Packt Publishing. <https://medium.com/analytics-vidhya/natural-language-processing-basic-concepts-a3c7f50bf5d3>
- Stajner, S. (2021). Automatic Text Simplification for Social Good: Progress and Challenges, 2637–2652. <https://doi.org/10.18653/v1/2021.findings-acl.233>
- Strubell, E., Ganesh, A. & McCallum, A. (2019). Energy and Policy Considerations for Deep Learning in NLP.
- Suleiman, D. & Awajan, A. (2020). Deep Learning Based Abstractive Text Summarization: Approaches, Datasets, Evaluation Measures, and Challenges. *Mathematical Problems in Engineering*, 2020.
- Suter, J., Ebling, S. & Volk, M. (2016). Rule-based Automatic Text Simplification for German.
- Swayamdipta, S. (2019, januari 22). *Learning Challenges in Natural Language Processing*. Verkregen 1 april 2022, van <https://www.microsoft.com/en-us/research/video/learning-challenges-in-natural-language-processing/>
- Tanya Goyal, G. D., Junyi Jessy Li. (2022). News Summarization and Evaluation in the Era of GPT-3. *arXiv preprint*.
- Thangarajah, V. (2019). Python current trend applications-an overview.
- Tops, W., Callens, M., Brysbaert, M. & Schouten, E. L. (2018). *Slagen met Dyslexie in Het Hoger Onderwijs*. Owl Press.
- Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E. & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models.

- van der Meer, C. (2022). Dyslexie hebben is Niet Zo Raar: Lezen is iets heel onnatuurlijks. <https://www.demorgen.be/beter-leven/dyslexie-hebben-is-niet-zo-raar-lezen-is-iets-heel-onnatuurlijks~bc608101/>
- Vasista, K. (2022). Evolution of AI Design Models. *Central Asian Journal of Theoretical and Applied Science*, 3(3), 1–4.
- Verhoeven, W. (2023, februari 8). *Applaus voor de studenten die ChatGPT gebruiken* (Trends, Red.). https://trends.knack.be/economie/bedrijven/applaus-voor-de-studenten-die-chatgpt-gebruiken/article-opinion-1934277.html?cookie_check=1676034368
- Verma, P. & Verma, A. (2020). A review on text summarization techniques. *Journal of scientific research*, 64(1), 251–257.
- White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J. & Schmidt, D. C. (2023). A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT.
- Xu, W., Callison-Burch, C. & Napoles, C. (2015). Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3, 283–297.
- Zeng, Q., Kim, E., Crowell, J. & Tse, T. (2005). A Text Corpora-Based Estimation of the Familiarity of Health Terminology. In J. L. "Oliveira, V. Maojo, F. Martín-Sánchez & A. S. Pereira (Red.), *Biological and Medical Data Analysis* (pp. 184–192). Springer Berlin Heidelberg.
- Zhang, M., Riecke, L. & Bonte, M. (2021). Neurophysiological tracking of speech-structure learning in typical and dyslexic readers. *Neuropsychologia*, 158, 107889.
- Zhou, W., Ge, T., Xu, K., Wei, F. & Zhou, M. (2019). BERT-based Lexical Substitution. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3368–3373. <https://doi.org/10.18653/v1/P19-1328>