Universiteit Utrecht

BERT: HET MODEL ONTMASKERD

Een beschrijvend literatuuronderzoek naar de effecten van ontwerpkeuzes omtrent de pre-training van taalspecifieke BERT-modellen

SAMENVATTING

De recente ontwikkelingen binnen Natural Language Processing zorgen voor nieuwe mogelijkheden voor onderzoeken naar betere modellen voor taalbegrip. Het in 2018 gepubliceerde model onder de titel Bidirectional Encoder Representations from Transformers (BERT) speelt een actuele rol in de verbetering van NLP. In dit literatuuronderzoek worden de aspecten van het model een voor een uitgelegd, geanalyseerd en vergeleken met andere, nieuwere versies van de BERT-architectuur. Hierbij wordt gekeken naar modellen die zijn getraind op verschillende talen, en in welk opzicht iedere versie van BERT zich onderscheid van de rest. Na de verschillen te hebben gekoppeld aan de resultaten op een paar NLP-taken blijkt dat een grote pre-trainingsdataset met een grote woordenschat cruciaal is voor het functioneren van het model. Ook blijkt de Whole-word masking taak een belangrijke rol te spelen. Het al dan niet toevoegen van een trainingstaak bovenop MLM geeft wisselende resultaten. Desalniettemin moet er nog nader worden bepaald wat de optimale configuraties zijn, door middel van verder onderzoek.

Tom Kalkman (6209890)

Bachelorscriptie BSc Kunstmatige intelligentie

7.5 ECTS

Begeleider: Rick Nouwen

Tweede Lezer: Gijs Wijnholds

Datum: 10 juli 2020

Inhoudsopgave

Inleiding	**1**
Hoofdstuk 1: BERT	**4**
Hoofdstuk 2: Pre-Training	**11**
Hoofdstuk 3: Fine-Tuning	**17**
Conclusie & Discussie	**22**
Bibliografie	**23**

Inleiding

Het verwerken van natuurlijke taal is al vanaf het begin van de ontwikkeling van computers een belangrijk doel geweest binnen het vakgebied van de kunstmatige intelligentie. Zo noemde Alan Turing het verwerken van taal al één van de hoofdcriteria voor intelligentie (Turing, 1950). *Natural Language Processing* (NLP) is bijvoorbeeld nodig wanneer je een systeem een actie wil laten uitvoeren volgens jouw instructies. Voor het merendeel van de bevolking is een programmeertaal vrijwel onbegrijpelijk. Een technologie die computers laat werken met natuurlijke taal zorgt ervoor dat mens en machine gemakkelijker met elkaar kunnen communiceren. Een spraakherkenningsgericht apparaat, zoals Amazon Echo of Siri, moet de klanken van een commando eerst analyseren en naar getallen omzetten alvorens te kunnen verwerken wat er van het apparaat gevraagd wordt. Andere toepassingen, zoals auto-correctie, hebben enkel geschreven tekst nodig om hun taak tot een goed einde te brengen.

Voordat een programma een tekst efficiënt en succesvol kan verwerken, heeft het een bepaalde representatie van taal nodig. Die representatie is tegenwoordig meestal in de vorm van een grote verzameling vectoren gevuld met reële getallen (Mikolov et al. 2013). De vectoren vormen een kansverdeling, die de waarschijnlijkheid van een woord uitrekent op basis van zijn context. Voor twee woorden als ‘koning’ en ‘koningin’ lijken hun individuele vectoren veel op elkaar, omdat ze veelal gebruikt worden binnen dezelfde context. Als een bepaald woord bijna nooit in de buurt van een ander specifiek woord voorkomt na een grote hoeveelheid tekst geanalyseerd te hebben, zullen de individuele vectoren voor die woorden meer afstand tot elkaar hebben, omdat de kans klein is dat deze woorden dichtbij elkaar staan op een semantisch niveau. De vectoren worden niet opgesteld door programmeurs, maar geleerd door het programma zelf en continu aangepast om de taalverbanden beter te simuleren. Uiteindelijk is het doel van een taalmodel om een dermate goede representatie van taal te hebben dat het zijn taak feilloos kan uitvoeren op een zo efficiënt mogelijke manier.

Om te testen hoe goed een programma kan omgaan met tekst, zijn er voor iedere aparte Natural Language Processing taak bepaalde testen ontwikkeld om een score toe te kennen aan de prestatie van een programma, zodat de uitvoeringen van verschillende modellen makkelijker te vergelijken zijn. Een op dit moment veel gebruikte test is de General Language Use Evaluation-benchmark (GLUE), een test die verschillende taken apart bekijkt en op basis van die scores een totaalscore berekent die moet weergeven in hoeverre een model correct en nauwkeurig werkt (Wang et al., 2018). In 2018 publiceerden Devlin et al. een model onder de naam *Bidirectional Encoder Representations from Transformers* (BERT) dat op het moment van publiceren hoger scoorde dan ieder ander model op 11

NLP-taken die onder andere deel uitmaken van GLUE, waaronder *Natural Language Inference* en *Question Answering*. Hoewel er intussen al een handvol modellen zijn gepubliceerd die een nóg hogere score hebben behaald, sommige daarvan adaptaties van BERT, zal ik hier de originele versie bespreken. Dat is interessant omdat deze de basis vormt voor de meeste van die nieuwe modellen.

Ten opzichte van eerdere taalmodellen heeft BERT een aantal nieuwe technieken. Zo wordt de focus niet meer gelegd op een *Long-Short-Term-Memory Recurrent Neural Network,* maar op *Multi-Headed Attention.* Deze architectuur bleek volgens Vaswani et al. (2017) betere testresultaten op te leveren. Sinds de publicatie van dat artikel zijn er een aantal modellen ontwikkeld die de focus op Attention leggen, maar BERT was de eerste die zijn manier van training zodanig aanpaste dat het mogelijk was om een dieper begrip van context te creëren.

Zoals bij elk taalmodel is het functioneren van BERT erg afhankelijk van de trainingsdata. Zelfs een geavanceerde techniek kan falen op ingewikkelde taken zoals Question Answering als er een tekort aan diversiteit is bij de trainingsinput. De originele versie van BERT is getraind op het gehele Engelstalige corpus van Wikipedia. Later hebben de onderzoekers ook een meertalige versie getraind, door het corpus uit te breiden met teksten van de 100 meest gebruikte talen op Wikipedia (Devlin et al., 2019). Sinds de originele publicatie van de meertalige BERT (Devlin et al., 2018), zijn er over de hele wereld onderzoekers geweest die een eigen eentalige adaptatie gemaakt hebben van het model. Om die modellen tot stand te laten komen, zijn er een aantal afwijkende keuzes gemaakt wat betreft trainingsdata en trainingstaken, die wellicht leiden tot andere resultaten. Zo maakten Polignano et al. (2018) een Italiaans model van BERT dat hoofdzakelijk is getraind op data van Twitter, en kozen Le et al. (2019) ervoor om bij de training een corpus te gebruiken dat meer dan 5 keer zo groot is als datgene wat gebruikt werd voor de originele BERT. Liu et al. (2019) maakten een aantal aanpassingen aan de trainingsprocedure van het model, die betere resultaten op bleken te leveren.

In vergelijking met de meertalige versie van BERT, presteren de meeste eentalige versies beter (De Vries et Al., 2019; Polignano et al. 2019). Als duidelijk wordt welke combinaties van type data en pre-trainingsprocedure goed werken voor verschillende talen en taken, kan daar wellicht rekening mee gehouden worden bij het opstellen van een volgende meertalige versie van BERT, in plaats van voor iedere taal data te verkrijgen van dezelfde bron (Wikipedia) zoals voorheen is gedaan. Bij een bron als Wikipedia zijn er tussen talen onderling grote verschillen in hoeveelheid data. De Franse of Italiaanse Wikipedia beschikt over een ruimere verzameling aan tekst dan bijvoorbeeld de Azerbeidzjaanstalige Wikipedia.[1]

Voor het vakgebied van kunstmatige intelligentie zou een meertalige BERT die even nauwkeurig werkt voor iedere taal een waardevolle bijdrage zijn. Wellicht kan zo'n model zelfs een uitkomst bieden voor maatschappelijke kwesties zoals laaggeletterdheid/analfabetisme. Voor iemand met een minder voorkomende moedertaal, zoals het Azerbeidzjaans, zijn doorgaans minder middelen beschikbaar dan voor een vaker voorkomende taal, zoals het Spaans. Meertalige NLP geeft kleinere talen dezelfde mogelijkheden als grotere talen. Eén programma voor 100 talen is namelijk efficiënter dan een apart programma voor ieder van die talen, zeker voor praktische toepassingen zoals een vertaalmachine of stukken tekst waar invloeden van verschillende talen in voorkomen. Bedrijven zijn eerder geneigd om weinig gesproken talen in hun systemen op te nemen als het niets kost. Gegeven dit belang ga ik kijken

[1] Engelse Wikipedia had 5.8 miljoen artikelen in december 2018, Azerbeidzjaanse Wikipedia slechts 142.000 https://stats.wikimedia.org/EN/TablesArticlesTotal.htm

naar de ontwerpkeuzes van een aantal eentalige BERT modellen, en proberen een verband te vinden tussen die keuzes en de prestaties van het model.

Dit brengt mij tot de volgende onderzoeksvraag:

Wat is de invloed van de ontwerpkeuzes rondom training op de prestaties van versies van het taalmodel BERT? *Een beschrijvend literatuuronderzoek naar de verschillen in aanpak tussen verschillende taalspecifieke versies van BERT.*

Dit wil ik onderzoeken aan de hand van een aantal deelvragen:

Voor ieder nieuw eentalig model moet er een eigen woordenschat worden opgesteld. Welke vorm van taalrepresentatie wordt doorgevoerd in de modellen? Wat is het verschil tussen de methoden en wat voor effect heeft dat op de training? Dit zal ik bespreken in het eerste hoofdstuk, samen met een verdiepende beschrijving van de binnenprocessen van BERT.

Ieder onderzoek gebruikt een ander corpus, omdat er telkens op een andere taal wordt getraind. Waar worden deze grote hoeveelheden tekst vandaan gehaald, wat voor motivatie wordt daarvoor gegeven en wat voor verschil maakt de grootte ervan voor de training?

Niet alle onderzoeken pre-trainen het model op dezelfde manier als het originele Engelse BERT model. Welke taken worden er gekozen om op te trainen en hoe staat dat in verband met de gekozen data en de prestaties van het model? Het tweede hoofdstuk zal zich richten op de verschillende manieren van training, met de focus op keuze van corpora en pre-trainingstaken.

Om BERT toe te passen op een bepaalde NLP-taak, zoals Natural Language Inference, moet het model eerst gefinetuned worden op die taak. Welke taken worden vooral gebruikt om het BERT-model te testen en op welke manier heeft het model ervoor gezorgd om een hogere score te bewerkstelligen dan de meertalige BERT? Welke NLP-taken krijgen de grootste prestatieverbetering door het gebruik van een eentalig model ten opzichte van het meertalige model? Is er daarbinnen nog verschil tussen talen onderling? In het derde hoofdstuk zal ik proberen inzicht te geven in het antwoord op deze vraag.

Aan de hand van deze deelvragen zal ik proberen inzicht te krijgen over verbanden tussen de manier van trainen, finetunen en evalueren en de uiteindelijke prestatie van het model. Omdat er voor elk model andere documenten zijn gebruikt om mee te trainen, finetunen en evalueren is het tamelijk lastig om een correcte kwantitatieve vergelijking te maken tussen verschillende talen. Desalniettemin wordt er op dezelfde taken getest en zijn de meeste taalspecifieke NLP-taken ook getest op de meertalige variant van BERT, die dus wel goed kwantitatief te vergelijken zijn.

In het laatste hoofdstuk zal ik mijn bevindingen samenvatten en een antwoord geven op de onderzoeksvraag die ik hierboven gesteld heb.

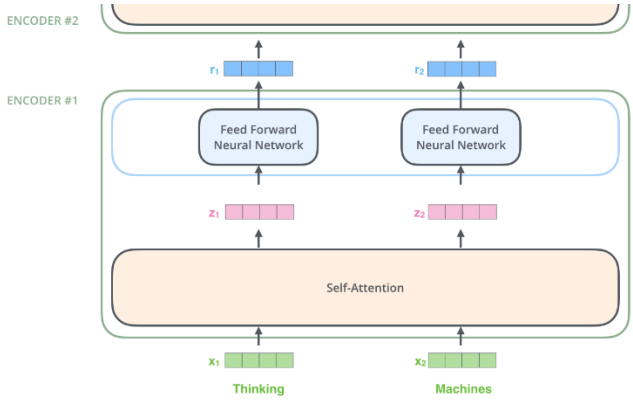
Hoofdstuk 1: BERT

1.1 Algemene samenvatting van BERT

BERT valt onder de categorie *pre-trained language models*. De ontwerpers van het model hebben gekozen voor de strategie finetunen bij het toepassen op specifieke taken. Dat wil zeggen dat er tijdens het pre-trainen zo min mogelijk taakspecifieke parameters worden geïntroduceerd en dat het voorgetrainde model zich op iedere taak apart moet afstemmen om deze succesvol te volbrengen. Dit is praktisch, omdat pre-training slechts eenmaal nodig is en het model vanaf dan een algemene taalrepresentatie heeft verworven. Het voordeel van een dergelijke constructie is dat onverwachts een nieuwe taak uitvoeren ongeveer evenveel tijd kost als ieder andere taak. Finetunen duurt gemiddeld minder dan een uur, terwijl de gehele pre-training van $BERT_{BASE}$ alleen al meerdere dagen nodig had (Devlin et al. 2018). Ik zal in deze sectie kort uitleggen hoe de architectuur van het model is opgebouwd. Vervolgens zal ik in volgende secties wat dieper in gaan op de individuele onderdelen van het BERT model.

Om te beginnen moet het model een bepaalde woordenschat hebben. Een groot corpus is daarvoor getokenized op zo'n manier dat de meest voorkomende woorden zijn opgenomen in de database. Alle overige woorden worden opgedeeld in hun meest voorkomende onderdelen. Zo wordt het weinig voorkomende woord 'embedding' in de originele BERT getokenized als: 'em', '##bed', '##ding'. De twee '#' symbolen voor de tokens geven aan dat deze tokens niet het begin van het woord zijn. Voor iedere token heeft het model een embedding; een vector bestaande uit 768 reële getallen. De embeddings worden zodanig getraind dat woorden die een sterke semantische overeenkomst hebben, dichtbij elkaar staan in de vectorruimte.

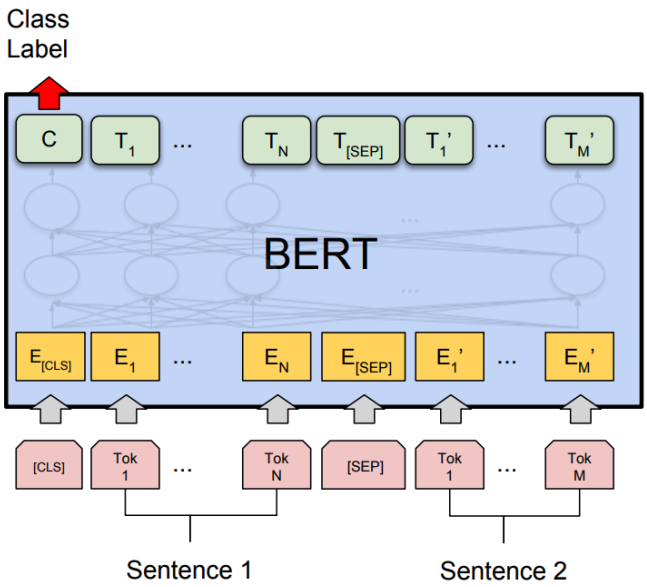
Nu er een basiswoordenschat is opgesteld, kan het model beginnen met leren. BERT leert taalkundige verbanden door twee ongesuperviseerde taken tegelijkertijd uit te voeren op een groot corpus. Net als voorgaande modellen maakt BERT hiervoor gebruik van de Transformer architectuur om de verbanden tussen woorden en zinnen te leren. Een Transformer is een type model dat lange-afstandsafhankelijkheden tussen input en output deduceert puur door middel van een aandachtsmechanisme en geen gebruik maakt van recurrence. De originele Transformer (Vaswani et al., 2017) bestond uit opeengestapelde encoders, die de tokens omzetten naar vectoren die de semantische en taalkundige constructies steeds beter weten te bevatten. In iedere encoder-laag zit een aandachtsmechanisme (attention) en een feed forward neuraal netwerk. Nadat er een getrainde representatie van kennis in het model is gestopt, worden de tokens door opeengestapelde decoders geleid, die de getrainde vectoren weer interpreteren tot natuurlijke taal. De verschillende versies van BERT maken gebruik van afwisselende aantallen lagen. Het $BERT_{LARGE}$ model bevat maar liefst 24 opeengestapelde lagen encoders, en de decoders zijn compleet geschrapt. Een decoder is nodig voor het omzetten van de in de encoders verkregen informatie over taal, terug naar (een andere) natuurlijke taal. Omdat de focus van BERT ligt op taalbegrip en niet op taalproductie is het ontbreken aan decoders logisch. Zie figuur 1 voor een abstract overzicht van een laag in de Transformer (Alammar, 2018).



Figuur 1 Originele Transformer. Herdrukt van "The Illustrated Transformer" door Alammar, J. 2018. Geraadpleegd van http://jalammar.github.io/illustrated-transformer/.

Het aandachtsmechanisme is de constructie waarbij de relatie tussen ieder individueel woord in de zin wordt onderzocht. Dit wordt self-attention genoemd. De functie van het feed forward netwerk is het verwerken van de aangepaste vectoren door ze onder andere te normaliseren en zodanig op te stellen zodat ze bij een volgende laag weer opnieuw door een attention layer kunnen gaan. De outputs van iedere encoder worden via de feed forward laag naar de volgende encoder gebracht, totdat er na een van tevoren gespecificeerd aantal lagen een geavanceerd model uit voortkomt.

Het unieke aan het BERT model is dat alle woorden kunnen worden vergeleken met alle andere woorden binnen de reeks, gebruikmakend van self-attention. Deze eigenschap wordt vaak aangeduid met de term bi-directional, of non-directional, omdat het aandachtsmechanisme niet met een bepaalde zinsvolgorde hoeft te werken. Dit werd mogelijk door het gebruik van de *Masked* Language Model (MLM) taak, doordat de woorden die moeten worden voorspeld, van tevoren worden gekozen en gemaskeerd. Bij eerdere soortgelijke modellen werd er gebruik gemaakt van de 'Language Model' taak, die steeds de zin afging en aan het programma vroeg welk woord er, gegeven de eerdere woorden in de zin, logisch zou volgen: Voor welk woord w_i uit de woordenschat is op plaats i $P(w_i|w_0 \dots w_{i-1})$ maximaal? Met de woorden erna werd geen rekening gehouden (Radford et al., 2018). Non-directional training zou bij de reguliere Language Model taak voor problemen zorgen, omdat een woord dat tijdens een trainingscycle voorspeld zou moeten worden, al als input gegeven werd bij een eerdere trainingscycle. Zo kan het model in zekere zin 'spieken'. Door alle tokens in acht te nemen, creëert BERT een dieper niveau van context voor het model.



Figuur 2 Abstracte illustratie van BERT. Herdrukt van "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", Devlin, J., 2018. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), page 4173

Naast het gebruik van het Masked Language model, leert het model ook verbanden op zinsniveau door middel van een Next Sentence Prediction (NSP) taak. Tegelijk met het uitvoeren van de maskeringstaak, moet het model ook aangeven of de twee stukken van de zin, gescheiden door een [SEP] token, logisch op elkaar volgen, of dat er een willekeurige zin achter is geplakt.

Na de pre-training heeft het model voldoende kennis opgedaan om getest te worden op enkele doorsnee NLP-taken. Alvorens getest te worden moet het vers gepre-trainde model nog finetunen op het toepassen van deze specifieke taak. Het fine-tune element van de training heeft vooral te maken met de vormgeving van de input en output en de focus van het model op bepaalde parameters. Zo is er voor een taak als Question Answering een andere input en output nodig dan voor een taak als Named Entity Recognition of Sentimentanalyse. De betreffende taak wordt gesuperviseerd uitgevoerd met behulp van een speciaal corpus. Voor Question Answering wordt het model gevoerd met vragen, korte teksten, en het echte antwoord. Het model moet dan binnen die tekst op zoek gaan naar het antwoord en past zich aan.

In de rest van dit hoofdstuk geef ik een wat diepere uitleg van de structuren van BERT, om te verduidelijken welke aspecten van het model een rol kunnen spelen in het ontwerp van nieuwe BERT-modellen, en wat die aspecten precies inhouden.

1.2 Tokenizer

Het representeren van woorden is een belangrijk onderdeel van een taalmodel. De manier van representatie die BERT gebruikt werd geïntroduceerd in 2013 als word embeddings in de vorm van vectoren. Deze waren gericht op gehele woorden, waardoor het lastig was om nieuwe, onbekende woorden te verwerken (Mikolov et al., 2013). Een handige oplossing daarvoor is om ook embeddings te maken voor ieder apart karakter. Echter, dezelfde combinaties van tekens komen zo vaak voor in allerlei verschillende contexten dat ze op zichzelf weinig betekenis hebben. Er kan dan niet goed

gekeken worden naar het grotere geheel. Vandaar dat de BERT modellen vooral gebruik maken van algoritmes die de meest voorkomende woorden en subwoorden verwerken tot vectoren op zo'n manier dat woorden die veel bij elkaar in een document staan ook soortgelijke vectoren hebben. Subwoorden vormen een handig midden tussen de algemene karakters en de specifieke gehele woorden, zodat er nieuwe woorden verwerkt kunnen worden zonder dat er weinig betekenisvolle verbanden worden gelegd tussen veel voorkomende tekens.

Om de juiste subwoorden op te nemen in de woordenschat van het programma, wordt er voor sommige versies van BERT gebruik gemaakt van een *Byte Pair Encoding Algoritme* (BPE). Allereerst neem je een grote hoeveelheid onbewerkte tekst, en je geeft aan hoe groot je woordenschat mag zijn, bijvoorbeeld 30K tokens. Aan het eind van ieder woord wordt er een teken '</w>' toegevoegd, zodat het algoritme weet wanneer een woord compleet is. Voordat de woordenschat gemaakt kan worden moet de tekst dus al getokenized zijn op spaties of een andere vorm van woordgrenzen. Vervolgens worden alle woorden opgesplitst in losse tekens. Dan wordt er gekeken welke combinatie van tekens het vaakst naast elkaar staat. Die twee worden voortaan gezien als één eenheid. In het begin zal de meerderheid van de meest voorkomende combinaties uit losse tekens bestaan die enkel worden samengevoegd tot koppels (bijv. 'h' en 'e' -> 'he'), maar naarmate het algoritme voortgaat zullen ook die koppels vaak naast andere tekens voorkomen en uiteindelijk worden samengevoegd tot de meest voorkomende gehele woorden ('t' en 'he' -> 'the'). Dit herhaalt zich net zo lang tot er een woordenschat is ontstaan van de aangegeven grootte.

De originele versie van BERT (Devlin et al., 2018) maakt gebruik van een subwoordrepresentatie-algoritme dat WordPiece heet (Wu et al., 2016). Het lijkt sterk op BPE, maar kijkt niet zozeer naar de frequentie van combinaties van tokens, maar WordPiece voegt tokens samen als de likelihood van die combinatie op dat moment het hoogst is. De likelihood wordt berekend door de kans om de individuele tekens tegen te komen af te trekken van de kans om de combinatie van tekens tegen te komen. Dus bijvoorbeeld: likelihood van de combinatie 'de' = P('de') - P('d' + 'e'). Op die manier worden juist de (sub)woorden die het model het best doen fitten aan de trainingsdata aan het model toegevoegd. Dus: als er tijdens het trainen gebruik wordt gemaakt van alleen maar wetenschappelijke corpora met betrekking tot een bepaald onderwerp (Lee et al. 2019), zal de gecreëerde lijst van bekende (sub)woorden en hun vectorrepresentatie er compleet anders uit zien dan wanneer er puur en alleen naar twitterdata gekeken is (Polignano et al. 2019).

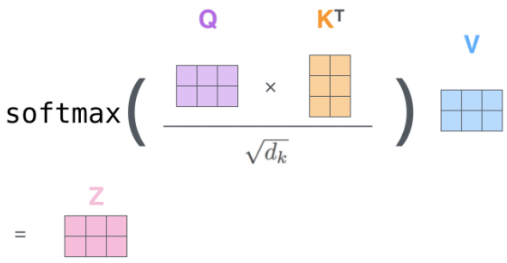
Nieuwere modellen van BERT maken gebruik van een andere versie van BPE, genaamd SentencePiece. Ook deze lijkt sterk op de voorgangers, maar verschilt op een fundamenteel niveau. BPE en WordPiece hebben het nadeel dat de input al van tevoren getokenized moet zijn, om ervoor te zorgen dat ze niet per ongeluk tekens samenvoegen die niet bij hetzelfde woord horen. Om te pre-tokenizen moeten er al keuzes worden gemaakt waarop er wordt getokenized, bijvoorbeeld door per spatie te splitsen. Echter, in niet alle talen worden woordgrenzen aangeduid met spaties, zoals bijvoorbeeld in het Japans of het Chinees. SentencePiece lost dit probleem op door alle tekens te coderen in Unicode, inclusief de spaties en leestekens.

1.3 Transformer-architectuur

Nadat de woordenschat van het model is opgesteld, is de volgende stap om het model te gaan pre-trainen. De inmiddels getokenizede tekst wordt in de Transformer gestopt in stukken van maximaal 512 tokens per keer. Er wordt niet zozeer gekeken naar volzinnen op een grammaticaal niveau, maar eerder naar een reeks van achtereenvolgende woorden die niet gelijk op hoeven te lopen met de grammaticale zinnen. De eerste token is altijd [CLS], en de reeks wordt gescheiden in twee stukken

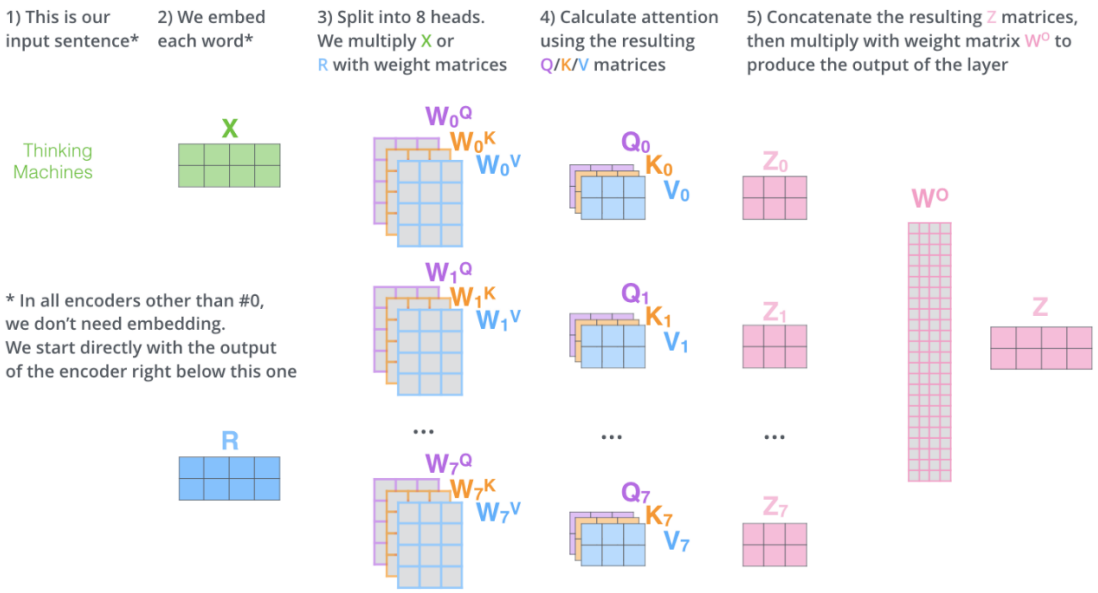
met de [SEP] token. De eerste helft van de tokens in de sequentie telt bovenop hun woordembedding een embedding die aangeeft dat deze tokens deel zijn van ‘zin A’, en de tweede helft, na de [SEP] ontvangt een embedding voor ‘zin B’. Ook wordt voor iedere token bijgehouden op welke plek in de zin deze staat, door middel van een positietag. Voor die positietag wordt gebruik gemaakt van een sinusoïde-functie die voor alle 512 mogelijke posities een unieke vector heeft. Met deze embeddings begint de pre-training in de eerste laag. De eerste laag bevat, net als alle andere lagen, twee achtereenvolgende onderdelen: een aandachtsmechanisme en een feed forward netwerk.

Het aandachtsmechanisme gaat als volgt te werk. We beginnen met de embeddings voor ieder individueel (sub)woord, die zijn verkregen door de tokenizer. De matrix met de woordembeddings wordt vermenigvuldigd met een aantal matrices die tijdens het tokenizen zijn meegetraind. Er zijn drie aparte matrices waarmee de embeddings worden vermenigvuldigd. Een ‘query’ matrix, een ‘key’ matrix en een ‘value’ matrix. Door te vermenigvuldigen met deze matrices, worden er drie nieuwe, kleinere matrices gecreëerd van 64 getallen lang, en even hoog als er woorden in de reeks stonden (dus max. 512). Om erachter te komen met welke woorden ieder woord het meest in contact staat, wordt de query matrix vermenigvuldigd met de key matrix. De uitkomst daarvan is een vierkante matrix, met op iedere rij en kolom een score voor een bepaalde combinatie van woorden. Hoe hoger de score, hoe sterker het verband tussen de woorden. De scores worden genormaliseerd om ze makkelijker te vergelijken en veranderd in een kansverdeling door middel van een softmax-functie. De softmax-score is de score die bepaalt in hoeverre er aandacht moet worden geschonken aan een woord in de zin. Die scores worden vermenigvuldigd met de value matrix zodat de waardes gewogen zijn. Een woord met een softmax-score van 0.001 zal nu minder focus ontvangen dan een woord met een softmax-score van 0.8. Op die manier probeert self-attention de relaties tussen woorden die weinig verband met elkaar hebben eruit te filteren, en juist extra aandacht te geven aan de relaties in de zin die wel belangrijk zijn. De uiteindelijke matrix die wordt doorgegeven aan het feed forward netwerk is de som van alle gewogen value vectoren. Zie figuur 3 (Alammar, 2018).



Figuur 3 self-attention calculation in matrix form. Herdrukt van “The Illustrated Transformer” door Alammar, J. 2018. Geraadpleegd van http://jalammar.github.io/illustrated-transformer/.

Echter, dit gebeurt per laag niet één keer, maar twaalf keer. Bij BERT is er namelijk sprake van Multi-Headed Attention. Hiermee kan het model zich focussen op verschillende plekken, en uiteindelijk er een gemiddelde uithalen. Er worden voor BERT dus tijdens het tokenizen twaalf verschillende query, key en value matrices random geïnitialiseerd en vervolgens getraind. Per ‘Attention Head’ worden de eindproducten aan elkaar geregen tot één langere vector, die weer 768 (12 * 64) getallen lang is. Als laatste wordt deze lange vector vermenigvuldigd met een laatste weight matrix, zodat de uiteindelijke output een enkele vector van niet al te groot formaat is. De matrix die uit deze laag komt, wordt dan naar de volgende laag gejaagd, zodat het model steeds kan voortbouwen op wat er al is ontdekt. Zie figuur 4 (Alammar, 2018) voor een geïllustreerd overzicht van de procedure.



Figuur 4 Multi-headed Attention gevisualiseerd. Herdrukt van "The Illustrated Transformer" door Alammar, J. 2018. Geraadpleegd van http://jalammar.github.io/illustrated-transformer/.

1.4 Pre-training

Om erachter te komen welke woorden met elkaar een verband hebben, moet er wel een vorm van training aan te pas komen. De reeksen worden niet zomaar door het model geleid, maar ze krijgen twee ongesuperviseerde taken: Next Sentence Prediction en Masked Language Model. De Next Sentence Prediction taak is bedoeld om het model een concept van zinsverbanden te geven. Dit is vooral handig voor taken waarbij er in meerdere zinnen moet worden gezocht naar een bepaald verband, zoals Question Answering. Het is aan het model om op basis van de informatie die is verzameld in de [CLS] token de zin na de [SEP] token te classificeren als 'IsNext' of 'NotNext'. Sommige stukken tekst worden door elkaar geschud, zodat BERT niet zeker weet of de tekst die het te zien krijgt logisch achtereenvolgend, of willekeurig naast elkaar geplaatst is. Die verdeling is 50-50. De keuze van BERT hangt af van de outputs die uit de Transformer komen, dus als er weinig verbanden gevonden zijn tussen de twee zinnen zal de kans dat de tweede zin willekeurig is hoger zijn. De reeks krijgt uiteindelijk het label met de hoogste softmax-score. Deze taak wordt gelijktijdig uitgevoerd met de MLM taak, dus krijgt de classifier ook de [MASK] tokens als input en niet de oorspronkelijke tokens.

Bij het uitvoeren van de Masked Language Model taak wordt verwacht van het model om te voorspellen welk woord er gestaan heeft, gebaseerd op de context. 15 procent van de tokens wordt aangepast. 12 procent wordt veranderd in de token [MASK], en het model moet daarvoor invullen welk woord er het meest waarschijnlijk zou zijn op basis van de kennis die wordt opgedaan door middel van het aandachtsmechanisme. Aan de overige 3 procent wordt een ander taakje verbonden. De helft van deze groep wordt vervangen door een ander woord wat er niet hoort, en de andere helft behoudt het woord wat er oorspronkelijk gestaan heeft. Echter, het model weet niet welke wel of niet verwisseld zijn. Het idee achter deze toevoeging is dat uitsluitend [MASK] tokens gebruiken niet per se leidt tot een goede voorspelling van de niet-gemaskeerde woorden, ook al worden ze wel gebruikt voor context. MLM is belangrijk omdat dit de taak is die ervoor zorgt dat het model niet maar één richting op wordt getraind, maar MLM laat het systeem toe om de linker en rechter context te fuseren. Zonder de gemaskerde woorden zouden de woorden zichzelf kunnen 'zien' omdat de tekst beide kanten op wordt verwerkt.

1.5 Fine-tuning

Nadat het model alle lagen van pre-training heeft doorlopen is het in principe klaar voor iedere taak. Echter, omdat BERT moet weten wat voor in- en output er per taak verwacht wordt, wordt er gefinetuned op een specifieke taak, door deze gesuperviseerd uit te voeren met behulp van een speciaal daarvoor opgesteld corpus. Ook wordt aan BERT duidelijk gemaakt welke vorm de input en de output krijgen. Voor elke input genereert BERT een output die wordt gecontroleerd door het gelabelde corpus. Zo leert het model op welke dingen er gelet moet worden, en kan het gebruikt worden voor een breed scala aan taken.

BERT is gefinetuned op 11 NLP-taken, waarvan negen deel uitmaken van de GLUE-benchmark. De GLUE-taken bestaan uit individuele zinsanalyses, parafraseertaken en inferentietaken. De belangrijkste, oftewel de taken waarvan de resultaten het vaakst worden getoond, zijn:

SST-2: Sentimentanalyse met behulp van de Stanford Sentiment Treebank. Het model moet een aantal zinnen uit filmrecensies classificeren als positief of negatief.

CoLA: Corpus of Linguistic Acceptability. Per zin moet er worden geëvalueerd of de zin wel of niet grammaticaal correct is.

STS-B: Semantic Textual Similarity. De data voor deze taak komt uit nieuwskoppen en video- en afbeeldingsonderschriften. De opdracht is om te beoordelen hoe gelijk de twee zinnen zijn aan elkaar, maar in plaats van een binaire beoordeling, wordt er een score tussen 1 en 5 toegekend.

MNLI: Multi-Genre Natural Language Inference. Er worden twee zinnen naast elkaar gezet. Een vormt de premisse en de ander de hypothese. De zinnen worden geclassificeerd als een *entailment* als de hypothese automatisch waar is wanneer de premisse geldt. De andere opties zijn een tegenspraak of neutraal. Voor deze opdracht wordt de data gehaald uit een breed scala aan bronnen.

De twee taken buiten de GLUE-benchmark zijn Question Answering en Reading Comprehension. Voor ieder van die taken is er getest hoe goed BERT presteert. Dit is gemeten met behulp van een Accuracy score, een F1-score of een Pearson Correlatie Coëfficiënt. In hoofdstuk 3 zal ik de taken bespreken die het vaakst worden getest en wat de pre-training voor effect kan hebben gehad op de resultaten.

1.6 Multilingual BERT, ALBERT en RoBERTa

Na de originele publicatie van BERT, zijn er een aantal onderzoekers geweest die erin geslaagd zijn om het model op verschillende punten te verbeteren. Zo hebben de makers van ALBERT (Lan et al. 2019) ervoor gezorgd dat het model beter presteerde met veel minder parameters en minder pre-training tijd, door de architectuur van de Transformer enigszins aan te passen. RoBERTa (Liu et al. 2019) liet de Transformerarchitectuur voor wat het was en richtte zich op welke manier de pre-trainingstaken geoptimaliseerd kunnen worden en voorzag het model van diversere en grotere hoeveelheden data. Deze strategie bleek goed te werken en is veel overgenomen door nieuwere, eentalige BERT-modellen. Multilingual BERT heeft dezelfde architectuur als de originele BERT, met betrekking tot de hoeveelheid parameters en pre-training, maar gebruikt trainingsdata van Wikipediapagina's in 104 talen. Hoewel er voor veel talen relatief weinig data aan het model gegeven is, lukt het toch om een redelijke representatie voor die talen op te stellen.

Aangezien er behalve deze drie grote veranderingen in de modelarchitectuur nog een aantal modellen zijn toegevoegd voor verschillende talen, ga ik in het volgende hoofdstuk in op de ontwerpkeuzes omtrent het pre-trainen van een BERT model.

Hoofdstuk 2: Pre-Training

2.1 Achtergrondinformatie

Modellen voor verschillende talen zijn onder andere lastig te vergelijken, omdat ieder onderzoek een compleet ander corpus gebruikt. Voor de Franse BERT kan er immers niet op Engelse Wikipedia teksten worden getraind. Waar ze wel op te vergelijken zijn, is de grootte van de input en het tekstgenre. Geldt: hoe groter hoe beter? Of weegt de kwaliteit van de corpora zwaarder? In dit hoofdstuk zal ik verschillende ontwerpkeuzes belichten met betrekking tot de pre-training van de modellen. Hoewel ik meer dan 30 verschillende modellen gevonden heb, zal ik er maar twaalf bespreken, omdat er alleen voor deze genoeg informatie te vinden was in overzichtelijke publicaties. De overige modellen kon ik slechts vinden als programma zonder openbare documentatie. Tevens lijkt er tussen de modellen die ik ga bespreken onderling genoeg diversiteit te zijn, zie tabel 1. Ik ga de modellen vergelijken op trainingsdata, zowel op de typen teksten als de hoeveelheid data. Vervolgens onderzoek ik welke verschillende (versies van) trainingstaken er worden gebruikt en wat voor motieven er daarvoor worden gegeven.

Een corpus is gedefinieerd als 'een verzameling van teksten'. Er zijn geen richtlijnen voor de grootte van die verzameling om er wel of niet het etiket corpus op te plakken. Een corpus kan bestaan uit de tekst van alle zeven Harry Potterboeken, maar ook uit de tekst van alle Engelse Wikipediapagina's die op het internet te vinden zijn. De ontwerpers van de BERT-modellen zorgen er veelal voor dat de gebruikte teksten representatief zijn voor de taal die BERT moet leren. Er zullen in een corpus dus niet alleen maar teksten voorkomen die maar één onderwerp behandelen, tenzij dat het uiteindelijke en enige doel is van het onderzoek. Finetunen op een taak als Question Answering is vrijwel onmogelijk als de pre-traindata bestaat uit onsamenhangende, subjectieve filmrecensies. Voor een brede representatie is een groot corpus handig, simpelweg omdat meer tekst grotere kans biedt op diversere tekst en dus ervoor zorgt dat het model in aanraking komt met een grotere variëteit aan woorden, onderwerpen, zinsstructuren en context in het algemeen.

Ieder onderzoek legt zijn focus op andere factoren binnen de pre-training, dus zal ik er maar een paar bespreken. Zie tabel 2 voor een overzicht van de gevonden factoren.

Model	**Taal**	**Referentie**
AraBERT	Arabisch	Antoun et al. 2020
Chinese BERT	Chinees (Traditioneel + Vereenvoudigd)	Cui et al. 2018
BERT	Engels	Devlin et al. 2018
RoBERTa	Engels	Liu et al. 2019
ALBERT	Engels	Lan et al. 2019
FinBERT	Fins	Virtanen et al. 2019
FlauBERT	Frans	Le et al. 2019
CamemBERT	Frans	Martin et al. 2019
ALBERTo	Italiaans	Polignano et al. 2019
Multilingual BERT	Meertalig	Devlin et al. 2019
RobBERT	Nederlands	Delobelle et al. 2019
BERTje	Nederlands	De Vries et al. 2019

Tabel 1: Overzicht van bekeken modellen

2.2 Woordenschat:

Grootte

BERT bevat na tokenizen een woordenschat van 30K bekende (sub)woorden. Veel andere modellen treden in zijn voetsporen door eenzelfde grootte te nemen voor hun woordenschat. Echter, sommige onderzoekers gebruiken een grotere woordenschat van 50K (sub)woorden. Door meer woorden een eigen embedding te geven, in plaats van hen op te delen in veelvoorkomende onderdelen, kan er een vollediger beeld worden gevormd van een taal. De uitschieter is de Italiaanse ALBERTo (Polignano et al. 2019), die een woordenschat gebruikt van 128K (sub) woorden. Hoewel nergens te vinden is waar deze keuze op is gebaseerd, vermoed ik dat dit komt door de variëteit aan typo's, hashtags en emoji's die worden gebruikt in de tweets. De Chinese versie (Cui et al. 2019) heeft als enige een woordenschat die kleiner is dan 30K. Ik vermoed dat dit te maken heeft met het verschil tussen een 'woord' en een 'Chinees karakter'. Een Chinees woord wordt doorgaans gevormd door een klein aantal karakters met een eigen betekenis, die in combinatie met andere karakters een nieuwe betekenis aannemen. Het Chinese woord voor brood wordt bijvoorbeeld gevormd door een combinatie van de karakters voor 'gezicht' en 'pakket'. Een karakter heeft in het Chinees dus meer de functie van een subwoord in het Engels. Een woordenschat van 21K (combinaties van) Chinese karakters bevat dus relatief meer informatie dan 30K Engelse subwoorden. Alleen van het Nederlandse model RobBERT (Delobelle et al. 2019) is niet bekend hoeveel woorden de woordenschat bevat, maar aangezien het niet wordt vermeld is het logisch om aan te nemen dat deze binnen de bekende marges valt (30K).

Tokenizer

Ook het algoritme dat wordt gebruikt om de onbewerkte teksten te tokenizen kan verschillen. WordPiece wordt bij verreweg de meeste modellen vervangen door SentencePiece, maar ondanks de voordelen van het algoritme, wordt de tekst vaak alsnog pre-tokenized. De Finse BERT maakt bijvoorbeeld onderscheid tussen een 'cased' versie, waarbij interpunctie is gescheiden en een versie waarbij alle karakters zijn omgezet in kleine letters zonder accenten. Voor ALBERTo is Ekphrasis geïmporteerd (Baziotis et al., 2017). Dit is een populair programma om Twitterdata bruikbaar te maken voor NLP. Het verwijdert gevoelige gegevens zoals URLs, e-mails, telefoonnummers, etc.

RoBERTa (Liu et al. 2019) gebruikt een eigen, aparte tokenizer, alvorens BPE toe te passen. De modellen die de RoBERTa-architectuur volgen doen hetzelfde (Le et al., 2019; Delobelle et al. 2019). BERTje (de Vries et al. 2019) maakt gebruik van SentencePiece, maar de woordenschat die daaruit voortkomt, wordt weer omgezet in WordPiece-formaat, zodat het programma verenigbaar blijft met de originele BERT.

In tabel 2 heb ik een overzicht gemaakt van de data die de modellen gebruiken. Het viel me op tijdens mijn onderzoek dat de publicaties hun waarnemingen niet op eenzelfde manier weergeven, en ervoor kiezen om sommige parameters wel of niet te benoemen. Vandaar dat tabel 2 niet volledig ingevuld kan worden.

Model (taal)	Grootte (tokens) // (woordenschat)	Grootte (GB)	Type	Pre-training
AraBERT (ARAB)	2.5B tokens // SentencePiece 60K	24GB	Nieuws	Static Whole word MLM + NSP
Chinese BERT (CHI)	13.6M regels // 21K	-	Wikipedia	Static Whole word MLM
ALBERT (EN)	3.3B tokens //SentencePiece 30K	16 GB	Boeken + Wikipedia	Static Subword MLM + SOP
BERT (EN)	3.3B tokens // WordPiece 30K	16GB	Boeken + Wikipedia	Static Subword MLM + NSP
RoBERTa (EN)	- // BPE 50K	160GB	Boeken, Wikipedia, Nieuws, Verhalen, OpenWebTekst	Dynamic Subword MLM
FinBERT (FIN)	3.3B tokens // SentencePiece 50K	-	Nieuws, Common crawl, Suomi24	Static Whole word MLM
FlauBERT (FR)	12.79B tokens // BPE 50K	71GB	Wikipedia, Boeken, Common Crawl	Dynamic Subword MLM
CamemBERT (FR)	32.7B tokens// SentencePiece 32K	138GB	OSCAR, non-shuffled	Dynamic Whole word MLM
AlBERTo (IT)	200M Tweets // SentencePiece 128K	191GB	Twitter	Static Subword MLM
Multilingual BERT	-//120K	-	Wikipedia	Static Subword MLM + NSP
BERTje (NL)	2.4B tokens // WordPiece 30K	12GB	Boeken, nieuws, Wikipedia, SoNaR-500	Static Whole (word) MLM + SOP
RobBERT (NL)	6.6B tokens //BPE	39GB	OSCAR, shuffled common crawl	Static Subword MLM

Tabel 2: Overzicht van de modellen, hun trainingsdata, woordenschat en manier van pre-training

2.3 Pre-training data:

Type:

De originele BERT (Devlin et al. 2018) gebruikt het BooksCorpus, gevuld met 800 miljoen woorden, samen met het Wikipedia corpus, dat bestaat uit 2,5 miljard woorden. Er is nadrukkelijk gekozen voor een corpus van document-niveau, bestaande uit zinnen die samen een coherent geheel vormen. Voor het ontdekken en leren van langeafstands-afhankelijkheden is het cruciaal voor het model om met langere, samenhangende zinnen te werken.

Samenhang

Toch is het niet zo dat deze structuur wordt gehandhaafd door ieder model. De Italiaanse AlBERTo (Polignano et al. 2019) is uitsluitend getraind op tweets, die op zichzelf staan en niet in direct verband met een ander deel van het corpus. Ook de Nederlandse RobBERT (Delobelle et al, 2019) is getraind op een corpus dat niet bestaat uit coherente documenten. De input voor dit model was het gigantische

meertalige OSCAR corpus (Ortiz Suárez et al., 2019), dat per regel door elkaar heen is geschud vanwege copyright redenen, waardoor er geen gebruik gemaakt kan worden van lange zinnen. Hoewel CamemBERT (Martin et al., 2019) ook het OSCAR corpus als input neemt vanwege de grootte, gebruikt dit model data die niet is gehusseld. De keuze voor een corpus zonder samenhangende zinnen is begrijpelijk, want deze zijn vooral van belang wanneer de Next Sentence Prediction taak uitgevoerd wordt en dat is bij zowel het Italiaanse als het Nederlandse model niet het geval. Voor de Masked Language Model taak maakt het geen verschil of de aaneensluitende zinnen wel of niet met elkaar te maken hebben, omdat iedere sequentie op zichzelf wordt bekeken.

Stijl en formaliteit

De overige modellen maken wél gebruik van corpora op document-niveau, ongeacht de keuze voor of tegen NSP. Naast boeken en Wikipedia is de volgende meest gebruikte bron van trainingsdata het nieuws. Voor die corpora wordt er van verschillende nieuwsbronnen alle tekst uit een bepaalde periode gebruikt. Het eerste Nederlandse model, BERTje (De vries et al. 2019), gebruikt precies die drie typen data. De Arabische versie van BERT (Antoun et al., 2020) is uitsluitend getraind op 24GB aan nieuwsartikelen. Het nadeel van de hiervoor genoemde drie bronnen is dat er vrijwel geen informele taalconstructies worden gebruikt, terwijl die in het dagelijks leven vrij prominent aanwezig zijn. Om die reden trainen sommige versies zich ook op een corpus dat bestaat uit teksten die op internet zijn geschreven door amateurtekstschrijvers, zoals blogs, fora of sociale media. Zo bestaat het grootste deel van de trainingsdata van de Finse BERT (Virtanen et al. 2019) uit discussies die zijn gevoerd op een online forum tussen 2001 en 2017. De onderwerpen, stijl en formaliteit van de discussies lopen erg uiteen en zijn daarom een goede aanvulling op de overige databronnen van dit model.

In de paper van het Italiaanse model wordt uitgelegd dat veel NLP taken voor het Italiaans worden uitgevoerd voor de analyse van sociale media, vandaar hun keuze voor tweets als trainingsdata. Het doel van AlBERTo was om het eerste Italiaanse model voor taalbegrip te zijn dat de taal van sociale media representeert. Het Nederlandse model BERTje heeft juist gefilterd op documenten die niet afkomstig waren uit chats of Twitter, vanwege kwaliteitsafwegingen.

Grootte data:

Met 16GB aan tekst is BERT's trainingsdata één van de kleinere tussen de modellen die ik bestudeerd heb. Alleen BERTje (De vries et al. 2019) heeft een kleinere trainingsdataset van 12GB aan tekst. Nadat RoBERTa (Liu et al. 2019) had aangetoond dat een grotere dataset een betere score opleverde op vrijwel alle NLP taken, was het niet meer dan logisch om uit te zoeken hoeveel data optimaal resultaat levert. Veel trainingsdata zorgt ook voor een langere trainingstijd, vandaar dat er na RoBERTa geen onderzoeken meer zijn gedaan naar nog grotere trainingsdata.

De totale hoeveelheid trainingsdata bevat bij ieder model tussen de 2.4 miljard en 32.7 miljard tokens. Alleen RoBERTa kan nog meer tokens verwerkt hebben bij het pre-trainen, maar de exacte hoeveelheid is niet bekend.

Het Franse model FlauBERT, en het Finse model hebben allebei agressief gefilterd op de data die ze hadden verkregen uit de internetbronnen, vanwege overlappende tekst, of tokens zonder betekenis van waarde zoals telefoonnummers of e-mailadressen. Hierdoor bleef er maar een relatief klein gedeelte van de originele tekst over. Desondanks bevat het gefilterde corpus evenveel tokens als de trainingsdata voor de originele BERT, en wel 30 keer de hoeveelheid van het Finse aandeel in de Meertalige BERT.

2.4 *Trainingstaken*:

NSP:

Niet alleen de gebruikte data verschilt, maar ook de manier van training kan afwijken. Het verwijderen van de Next-Sentence Prediction taak is voor de originele BERT (Devlin et al., 2019) getest, met de overige parameters gelijk. Op alle geteste taken presteerde het model zonder NSP slechter. Echter, andere onderzoeken wijzen op het tegenovergestelde. In het onderzoek van RoBERTa (Liu et al. 2019) bleek dat het verwijderen van de NSP-taak vooral de prestaties van het model schaadt, wanneer de sequenties die aan het programma gegeven worden geen volzinnen zijn, maar op willekeurige plekken zijn afgesneden, zoals het geval is bij de originele BERT. RoBERTa heeft aangetoond dat het gebruik van volzinnen, zonder NSP de hoogste resultaten oplevert. De NSP-taak bleek vooral te focussen op onderwerpovereenkomsten, en is daarom te makkelijk. Voor iedere zin die het label 'NotNext' zou moeten krijgen, is de kans groot dat deze zin over een compleet ander onderwerp gaat. Voor het model is het hierdoor relatief makkelijk om de zinnen binair te classificeren en leert het niet de coherentie tussen twee zinnen herkennen zoals aanvankelijk bedoeld was. Een groot deel van de modellen die ik bestudeerd heb, treedt in de voetsporen van RoBERTa en laat de NSP-taak weg.

Voor ALBERT (Lan et al. 2019) wordt er gebruikgemaakt van een andere taak die op Next Sentence Prediction lijkt: Sentence Order Prediction (SOP). Deze nieuwe taak kijkt of twee segmenten in de juiste volgorde staan of dat ze andersom moeten staan. Gegeven dat twee aaneengrenzende zinnen vaker over hetzelfde onderwerp gaan, is dit een relatief moeilijkere taak. Let wel dat ALBERT voor deze taak geen volzinnen gebruikt, maar willekeurig afgesneden segmenten, net als de originele BERT. Zo wordt het model geforceerd om fijnere, distinctieve eigenschappen te leren over coherentie. BERTje is het enige andere model dat ik bestudeerd heb die deze taak overgenomen heeft.

MLM:

Ook de Masked Language Model taak wordt niet voor ieder model op dezelfde manier uitgevoerd, ook al wordt deze taak in geen enkel model dusdanig veranderd dat hij niet meer terug te herkennen is. MLM is een vast onderdeel van BERT, en in een aantal modellen is de taak enigszins aangepast.

Dynamic masking

RoBERTa was het eerste model om de MLM taak aan te passen, door te onderzoeken of de maskeringstaak verbeterd kon worden door gebruik te maken van dynamic masking, waarbij er per trainingscycle een nieuwe subset van een reeks wordt gemaskeerd, in plaats van static masking, waarbij er 10 kopieën worden gemaakt van de trainingsdata om 10 verschillende versies van de gemaskeerde data te verkrijgen. Iedere trainingscycle zullen echter wel dezelfde [MASK] tokens terugkeren. Bij dynamic masking is er dus sprake van meer diversiteit. Deze versie van MLM wordt alleen overgenomen door de Franse modellen.

Whole word Masking

Ongeveer de helft van de versies van BERT gebruikt Whole Word Masking. Dat wil zeggen dat een gemaskeerd subwoord niet meer geïsoleerd kan voorkomen, maar dat alle andere onderdelen van dat woord ook gemaskeerd zullen zijn. Voor het model is dit uitdagender dan slechts een veelvoorkomend WordPiece token proberen te achterhalen, zeker als deze omringd is door zichtbare onderdelen die tot hetzelfde woord behoren. Vooral voor talen zoals het Fins, of Japans waarin er veel gebruik wordt

gemaakt van agglutinatie, het samen- en toevoegen van verschillende morfemen om de rol van een woord in een zin duidelijk te maken, is deze toevoeging van belang. Zo heeft het woord voor ‘vrouw’ in het Fins twaalf verschillende vormen, die afhankelijk zijn van de functie van het woord in de zin.

2.5 Algemene Architectuuraanpassingen

ALBERT is het enige model wat ik ben tegengekomen dat de Transformer-architectuur enigszins heeft aangepast. Ten eerste is de grootte van de vectoren in de hidden layer niet langer gelijk aan de grootte van de woordembeddings. ALBERT laat de embeddings geprojecteerd worden op twee kleinere matrices, die vervolgens weer tot een hidden layer wordt omgerekend. Vocabulary x Hidden Layer geeft namelijk een veel groter getal dan Vocabulary x Factorized Embedding + Factorized Embedding x Hidden Layer als H >> E. Dit verkleint het aantal gebruikte parameters en zorgt ervoor dat het model sneller gepre-traind kan worden (Lan et al. 2019).

Ten tweede delen alle twaalf lagen van de Transformer dezelfde parameters, in plaats van twaalf verschillende, zoals het geval was voor BERT (Devlin et al., 2018). Hoewel deze keuze zorgt voor enig verlies met betrekking tot de prestaties, is het aantal parameters drastisch gedaald, en daarmee is de snelheid van het model gestegen.

Nu ik alle bestaande aanpassingen heb belicht, ga ik onderzoeken op welke manier die aanpassingen voor verbetering zorgen, en hoe dat getest wordt.

Hoofdstuk 3: Finetunen en Testen

3.1 Verschillende manieren van finetunen

Om BERT toe te passen op een bepaalde NLP-taak, zoals Natural Language Inference, moet het model eerst gefinetuned worden op die taak. Finetunen vindt plaats door het gepre-trainde model een gelabeld corpus te geven, dat is opgedeeld in zinnen of paren van zinnen. Het model geeft zelf een output, en die wordt vergeleken met het label dat al bij de input zat. Op deze manier kan BERT zichzelf verbeteren, en klaarstomen voor de ongelabelde testsets.

Welke taken worden vooral toegepast bij het testen van de modellen en op welke manier hebben de eentalige modellen ervoor gezorgd om een hogere score te bewerkstelligen dan de meertalige BERT? In ten minste tien eentalige BERT-modellen wordt er gefinetuned op Sentimentanalyse. Ieder met een andere dataset, vanwege taaldiversiteit, maar wel allemaal geëvalueerd met een accuracy-score en een verschil ten opzichte van de meertalige BERT (Nozza et al., 2020). In dit hoofdstuk ga ik, na een korte bespreking van de manier van finetunen en de taken waarop gefinetuned wordt, in op de resultaten van verschillende BERT-modellen die gefinetuned zijn op de twee meest voorkomende taken.

De NLP-taken waar de BERT-modellen op gefinetuned en getest zijn, kunnen worden verdeeld over vier categorieën:

a. Sentence Pair Classification **MNLI**, QQP, QNLI, STS-B, MRPC, RTE, SWAG
b. Single Sentence Classification tasks**:** **SST-2,** CoLA
c. Question answering tasks: **QA**
d. Single sentence tagging tasks: **NER**, POS

Van iedere categorie zal ik de taak bespreken die het meest voorkomt bij de verschillende modellen. Voor sommige van de overige taken die in de GLUE-benchmark zijn opgenomen heb ik in sectie 1.5 een kort overzicht gemaakt.

Sentence Pair Classification is de categorie met de grootste diversiteit aan taken. Hoewel in alle gevallen twee zinnen met elkaar worden vergeleken, en daarna geclassificeerd, is het type classificatie steeds anders. De meest voorkomende taak, Natural Language Inference, wordt uitgevoerd door alle Engelse modellen, maar ook door de Chinese, en allebei de Franse versies van BERT. Het doel van de taak is om de tweede zin correct te classificeren als een logisch gevolg van de eerste, een tegenspraak met de eerste, of als neutraal. Voor alle niet-Engelse modellen wordt het passende deel van het meertalige XNLI-corpus gebruikt. Deze zijn voor iedere taal even groot, 7500 paren van zinnen, omdat de Engelse versie steeds door een machine is vertaald. De testsets zijn voor iedere taal handmatig vertaald. Vanwege het homogene aspect van de XNLI-dataset, zijn de resultaten voor deze taak beter te vergelijken.

Single Sentence Classification. Voor een classificatie verzamelt het model van alle tokens in de zin informatie in de voorste token: [CLS]. Voor een taak als Sentimentanalyse dragen alle componenten in de zin bij aan het positieve of negatieve sentiment wat uiteindelijk wordt toegekend aan de zin. Alle tokens worden geassocieerd met een bepaalde positieve of negatieve waarde, en vervolgens in de context van de zin geplaatst. Het sterk positieve zinsdeel ‘erg goed’ wordt bijvoorbeeld geclassificeerd als negatief wanneer het woordje ‘niet’ ervoor wordt geplaatst. Als inputdata wordt veelal recensies van boeken, films of hotels gebruikt, en in de Arabische en Italiaanse modellen ook tweets.

Single Sentence Tagging Tasks worden gezien als de relatief makkelijke NLP-taken, omdat ze niet veel semantische context nodig hebben om correct uitgevoerd te worden. Named Entity Recognition is de

taak binnen deze categorie die het vaakst voorkomt onder de modellen, hoewel de originele BERT zich er niet op heeft geëvalueerd. Het model moet zoveel mogelijk 'genoemde entiteiten' opsporen en classificeren. De categorieën zijn meestal: namen, locaties, organisaties en overige. Op de tweede plek komt Part of Speech tagging, waar de eerste BERT zich ook niet mee bezig heeft gehouden. Alle tokens krijgen bij deze taak een label dat past bij hun functie in de zin: '*noun', 'verb',* etc.

Question Answering Tasks. Dit is een categorie waartoe slechts één taak behoort, maar het is een unieke taak, omdat het model meer vrijheid heeft bij het genereren van een output. Er is geen sprake van een classificatie of een benoeming, maar het model moet zelf zoeken wat het antwoord kan zijn op een vraag. Weliswaar gebeurt dat binnen een relatief kleinschalig frame, maar het stuk tekst waarin het antwoord zich kan bevinden biedt zeker meer keuzes dan de mogelijkheden bij een binaire classificatie of Part-of-speech tagging. Hoewel de originele BERT relatief veel aandacht besteedt aan deze taak in hun paper, verwerken alleen de twee andere Engelse modellen: ALBERT en RoBERTa en het Arabische model AraBERT Question Answering in hun analyse. Dit is waarschijnlijk te wijten aan het gebrek aan taal- en taakspecifieke datasets, zoals SQuAD1.1 (Rajpurkar et al., 2016). Pas na de publicatie van de vele taalspecifieke BERT-modellen, is er voor de Franse taal een corpus opgesteld dat soortgelijk is aan SQuAD1.1 (d'Hoffschmidt et al., 2020). In de paper wordt uitgelegd dat de opkomst van eentalige taalmodellen binnen NLP de aanleiding voor het opstellen van zo een corpus was.

Omdat Natural Language Inference en Sentimentanalyse de twee meest breed-geteste NLP-taken zijn, heb ik ervoor gekozen om alleen deze twee verder te analyseren.

3.2 Natural Language Inference

Natural Language Inference wordt voor alle niet-Engelse modellen gefinetuned met hetzelfde corpus, vertaald naar de juiste taal. Voor de Engelse modellen worden er verschillende NLI-taken getest, zoals MNLI en QNLI. Omdat de MNLI-taak het meest lijkt op de XNLI-taak heb ik ervoor gekozen om alleen de resultaten van die twee taken mee te nemen in het onderzoek, zodat ik zo veel mogelijk vergelijkbare data heb. Er is gemeten hoe vaak de verschillende modellen juist konden voorspellen of de twee zinnen die met elkaar worden vergeleken, een logisch gevolg van elkaar zijn, elkaar tegenspreken of geen van beide. In tabel 3 heb ik een aantal modellen onder elkaar gezet, om de vergelijking enigszins compleet te maken.

Model	XNLI-Accuracy	Parameters
mBERT (Frans)	76.9	175M
CamemBERT-Base	82.5	110M
CamemBERT-Large	**85.7**	335M
FlauBERT-Base	80.6	138M
FlauBERT-Large	83.4	373M
Chinese BERT	77.8	110M
Chinese wwm-BERT	79.0	110M

Tabel 3: vergelijking van XNLI-resultaten.

CamemBERT behaalt een hoger score dan FlauBERT voor beide groottes (Le et al., 2019). De large-versies presteren in alle gevallen beter dan de base-versies, net als voor de originele BERT (Devlin et al. 2018). Een hoger aantal hidden layers is dus voordelig, net als een hoger aantal lagen. Een groot verschil in aanpak tussen CamemBERT en FlauBERT is dat de laatste op een twee keer zo kleine dataset

is getraind. De resultaten van NLI komen overeen met de notie dat een grote trainingsdataset waardevol is voor een goed werkend model.

Een ander verschil tussen de twee Franse modellen is dat CamemBERT is getraind met Whole Word Masking, en FlauBERT niet (Martin et al. 2019). Ik vermoed dat deze keuze een van de redenen is voor de hogere scores van CamemBERT. Dit geldt wel voor de twee Chinese modellen (Cui et al, 2019). Deze hanteren dezelfde architectuur, zijn getraind op dezelfde data en gefinetuned op dezelfde manier. Het enige verschil tussen de twee modellen is de Whole-word masking strategie, in contrast met de MLM-taak zonder rekening te houden met gehele woorden. De resultaten van het model met Whole-word masking zijn hoger, en ik vermoed dat deze strategie niet alleen gunstig is voor een Chinees model, maar ook voor andere talen. De resultaten van CamemBERT ondersteunen dit vermoeden. Verder heeft het model met de grootste trainingsdata hier de beste score, maar niet het model met de meeste parameters. Ik denk dat de grootte van de trainingsdata daarom een belangrijkere rol speelt bij het opstellen van een goed taalmodel, dan de hoeveelheid hidden layers of andere architectonische details.

In tabel 4 heb ik een aantal Engelse BERT-modellen onder elkaar gezet, met hun bijbehorende MNLI-score. Behalve ALBERT-base, een model met maar 1/9 van de hoeveelheid parameters in de originele BERT, halen alle modellen BERT-base en -large flink in. RoBERTa is in deze lijst de enige met een grotere trainingsdataset (Liu et al., 2019). Voor de niet-Engelse modellen is in het verschil tussen de scores van de meertalige BERT en de eentalige BERT te concluderen dat een grotere trainingsdataset de NLI-score verhoogt, en de hoge score van RoBERTa ondersteunt deze hypothese. Echter, de hoge score kan mede te danken zijn aan een andere component van het model, zoals de grote woordenschat, de toepassing van Dynamic masking of het weglaten van de NSP-taak.

De andere modellen in tabel 4 maken allemaal gebruik van dezelfde, relatief kleine dataset. Ondanks die gelijke variabele, presteren de grote ALBERT-modellen beter dan BERT-large. Niet omdat ALBERT een groter model is, want BERT-large heeft meer parameters, maar omdat er langer en efficiënter is getraind op de trainingsdata. Net als voor de niet-Engelse modellen presteren de grootste modellen niet per se het beste, maar spelen variabelen zoals hoeveelheid trainingsdata een grotere rol.

Ook een efficiënte architectuur, zodat er lang getraind kan worden, blijkt zijn vruchten af te werpen. Zelfs na 1 miljoen trainingsstappen wordt de trainingsdata van ALBERT niet overfit, dus is er een model getraind voor 1.5 miljoen trainingsstappen en deze bleek nog beter te presteren (Lan et al, 2019). RoBERTa presteert bijna even hoog als ALBERT op een taak waarbij het verband tussen twee zinnen wordt onderzocht. Dit is interessant, omdat RoBERTa zich alleen met MLM heeft getraind, en ALBERT ook met Sentence Order Prediction, een moeilijkere versie van NSP. De toegevoegde waarde van een Sentence Pair Classification taak (NSP of SOP) bij het pre-trainen is daarom vermoedelijk minder groot dan Devlin et al. (2018) rapporteerde. Een model dat een combinatie van de trainingsdata van RoBERTa en de architectuur en trainingstaken van ALBERT verwerkt, zou voor duidelijkheid kunnen zorgen over dit vraagstuk.

Model	**MNLI-accuracy**	**Parameters**
BERT-base	84.5	108M
BERT-large	86.6	334M
RoBERTa	90.2	**355M**
ALBERT-base	81.6	12M
ALBERT xx-large 1M	90.4	235M
ALBERT xx-large 1.5M	**90.8**	235M

Tabel 4: vergelijking van MNLI-resultaten

3.3 Sentimentanalyse

Voor sentimentanalyse zijn er, net als bij NLI, voor de Engelse modellen steeds dezelfde corpora gebruikt om op te finetunen en testen. De Stanford Sentiment Treebank hoort ook bij de GLUE-benchmark en bestaat uit filmrecensies, met bijbehorende aantekeningen van hun sentiment. Echter, er is geen meertalig corpus voor deze taak, zoals die er voor NLI wel is. De gebruikte corpora voor de niet-Engelse modellen behelzen hoofdzakelijk boekrecensies, en alleen in het geval van de Italiaanse, Chinese en de Arabische BERT-modellen, van tweets of andere sociale media.

Model	**SA accuracy**
BERT-Base	92.8
BERT-Large	93.7
RoBERTa	96.4
ALBERT xx-large 1.5M	96.9

Tabel 5: Engelse modellen en hun sentimentanalyse scores

In tabel 5 heb ik de resultaten van de Engelse modellen onder elkaar gezet. Net als bij NLI trekken RoBERTa en ALBERT aan het langste eind. De twee modellen behalen een score die niet ver van elkaar vandaan ligt, ondanks hun architectonische verschillen. De resultaten op Sentimentanalyse ondersteunen de notie dat beide ontwerpen waardevol zijn om nader te onderzoeken.

Model	**Accuracy**
CamemBERT	92.3
FlauBERT	93.1
mBERT (French)	86.15
mBERT(Dutch)	89.1
BERTje	93.0
RobBERT	**94.4**
AraBERT	86.7
mBERT (Arabic)	83.0

Tabel 6: Accuracy scores voor sentimentanalyse van boekrecensies

In tabel 6 heb ik een overzicht gemaakt van alle niet-Engelse modellen, die op een boekrecensie-corpus zijn gefinetuned. Behalve AraBERT, scoren alle eentalige modellen in de buurt van of zelfs hoger dan BERT-base. In tabel 6 kan je zien dat de Nederlandstalige RobBERT de hoogste score krijgt voor het inschatten van het sentiment van boekrecensies. Echter, de meertalige BERT scoorde ook al hoger op de Nederlandse taak dan op de Franse of Arabische. Het kan dus zijn dat de Nederlandse taal of het gebruikte corpus een voorsprong geeft op de taak. Na het Nederlandse model volgen de twee Franse modellen, en BERTje, het andere Nederlandse model. Voor deze taak presteert FlauBERT juist beter dan CamemBERT, terwijl dat voor NLI juist andersom was.

Ieder eentalig model verslaat zijn meertalige tegenhanger met ten minste 3.7 procentpunt, en FlauBERT zorgt voor een verbetering van bijna 7 procentpunt. Een overvloed aan trainingsdata maakt ook voor sentimentanalyse een groot verschil in prestatie. Voor Arabische BERT was het grootste verschil in aanpak met de meertalige versie een bijna zes keer grotere dataset om op te trainen (Antoun et al., 2020). Ook de daaruit voortgekomen woordenschat is in het eentalige model enorm opgeschaald. De overige aspecten van het model zijn gelijk aan die van de meertalige BERT. Gegeven de relatief grote scoreverbetering is het logisch om aan te nemen dat een grote trainingsdata en woordenschat het model ten goede beïnvloeden. De andere eentalige modellen bevestigen deze gedachtegang.

Model	Accuracy
ALBERTO	87.5
Arabert (Jordaans)	93.8
AraBERT(Levantijns)	59.4
AraBERT (Modern Standaard Arabisch /Egyptisch)	92.6

Tabel 7: Accuracy-scores voor sentimentanalyse van tweets

In tabel 7 heb ik de scores van een aantal niet-Engelse modellen die niet op boek- of filmrecensies zijn gefinetuned, maar op zinnen die van sociale media zijn gehaald, voornamelijk tweets. De Italiaanse ALBERTO is ook gepre-traind op tweets en heeft de grootste pre-trainingsdataset van alle modellen die ik bekeken heb (Polignano et al. 2019), maar een voordeel hiervan is niet direct terug te zien in de score. Hoewel het bijna niet mogelijk is om deze resultaten kwantitatief te vergelijken, vanwege afwijkende datasets, is wel duidelijk dat de grootte van de trainingsdata niet de enige component is die modellen als RoBERTa hoog heeft laten scoren.

De Arabische BERT heeft sentimentanalyse toegepast op drie verschillende corpora, geschreven in verschillende dialecten. Zoals in tabel 7 te zien is, was het model minder goed voorbereid op het Levantijnse dialect, maar juist wel op de Jordaanse en Egyptische dialecten. Omdat het Arabische model uitsluitend op nieuwscorpora is gepre-traind, kan de lage score op het Levantijnse corpus duiden op een ongelijke representatie van het Levantijnse dialect in de pre-trainingsdata. Dit zou betekenen dat diversiteit aan genres in pre-trainingsdata wel degelijk belangrijk is.

Gemiddeld genomen zijn de accuracy-scores voor sentimentanalyse een stuk hoger dan die voor NLI. Het zou dan ook interessant zijn om een diepere analyse te maken van prestaties van BERT-modellen op meerdere NLP-taken. Op dit moment zijn er nog weinig taken die voor veel talen worden geïmplementeerd, vanwege een gebrek aan gelabelde datasets om op te finetunen.

H4 Conclusie & Discussie

In mijn literatuuronderzoek ben ik tot de conclusie gekomen dat de ontwerpkeuzes rondom training een grote invloed hebben op de prestaties van BERT, maar niet altijd op dezelfde manier. RoBERTa en ALBERT hebben bijvoorbeeld een compleet andere aanpak, maar komen toch op nagenoeg gelijke scores uit. Ik heb niet één perfecte strategie kunnen vinden. Het is bijvoorbeeld niet zo dat een groter model altijd beter presteert, als er andere aspecten van het model niet optimaal zijn. Toch heb ik wel een aantal dingen ontdekt.

De anderstalige modellen lopen vaak aan tegen een gebrek aan testcorpora. Daarom zijn er weinig modellen gepubliceerd met een gemiddeld resultaat over een breed scala aan NLP-taken, zoals dat bij de Engelse variant wel kan. Samen met de opkomst van de eentalige BERT-modellen, komen er ook steeds meer testcorpora, zoals XNLI en een Franse versie van SQuAD.

Tijdens mijn onderzoek merkte ik dat er in weinig papers dezelfde aspecten van het gepresenteerde taalmodel worden belicht. Ook wordt er weinig uitleg gegeven over de motivatie achter de keuzes van de ontwerpen. Dit maakte het voor mij moeilijk om een vergelijking te maken tussen de modellen. Vooral de weergave van de gebruikte pre-trainingsdata moet worden gestandaardiseerd. 6GB aan data staat niet altijd gelijk aan een bepaald aantal tokens. Als toekomstige onderzoekers alle variabelen van hun taalmodel met een motivatie overzichtelijk in hun paper zouden zetten, zouden afwijkende keuzes en hun resultaten sneller opvallen en kan een volgend onderzoek zich focussen op relevantere aanpassingen.

Een aantal variabelen heeft tot nu toe zonder uitzondering hoog gescoord. De beste tokenizer is SentencePiece, vanwege zijn universele manier van woorden opbreken. Whole-word masking is in alle gevallen beter gebleken dan sub-word masking, en een grote trainingsdataset met een grote woordenschat werkt beter dan een kleine. Hoe groot die datasets en woordenschat precies moeten zijn is nog onduidelijk, maar behalve de computationele complicaties die kunnen optreden bij een groot model is er nog geen aanwijzing geweest dat er een maximum is aan de grootte van de datasets, waarbij de prestatie optimaal is.

Voor de overige variabelen is er nog geen bewijs dat ze de prestaties van het model verbeteren. Voor een weloverwogen keuze tussen Dynamic tegenover Static masking heb ik nog niet genoeg bewijs gevonden. Ook om uit te sluiten of NSP/SOP een belangrijke rol speelt in het leren van lange-afstandsafhankelijkheden moet er nog meer onderzoek worden gedaan. RoBERTa heeft geconcludeerd dat het toepassen van NSP de prestaties van het model schaadt, maar naar de effecten van Sentence Order Prediction is nog weinig onderzoek gedaan.

Een volgend meertalig BERT-model zou zich dus moeten focussen op het opnemen van grotere pre-training datasets voor iedere taal en per taal moet de woordenschat worden uitgebreid. Ook moet Whole-word masking als trainingstaak worden opgenomen. De overige aspecten van het model moeten nog nader worden onderzocht. Gegeven de grootte en complexiteit van een dergelijk model, zullen efficiëntie en trainingssnelheid voorop staan, anders zullen de kleinere talen wederom het onderspit moeten delven.

Bibliografie:

Alammar, J. (2018). The Illustrated Transformer [Blog post]. Geraadpleegd van from https://jalammar.github.io/illustrated-transformer/

Antoun, W., Baly, F. andHajj, H. (2020). AraBERT: Transformer-based model for Arabic language understanding. *arXiv preprint arXiv:2003.00104.*

Baziotis, C., Pelekis, N. and Doulkeridis, C. (2017). Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017*), pages 747–754, Vancouver, Canada, August. Association for Computational Linguistics.

Cui, J., Che, W., Liu, T., Qin, B., Yang, Z., Wang, S. and Hu, G. (2019). Pre-training with whole word masking for Chinese BERT. *arXiv preprint arXiv:1906.08101.*

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* pages 4171–4186.

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G., and Nissim, M. (2019). Bertje: A Dutch Bert model. *arXiv preprint arXiv:1912.09582.*

Delobelle, P., Winters, T., and Berendt, B. (2020). Robbert: a dutch roberta-based language model. *arXiv preprint arXiv:2001.06286*

Kudo, T. and Richardson, J. (2018) Sentence-piece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert: A lite bert for selfsupervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Le, H. Vial, L. Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L. and Schwab, D. (2019). FlauBERT: Unsupervised language model pre-training for French.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., Ho So, C., Kang J. (2019) BioBERT: a pre-trained biomedical language representation model for biomedical text mining

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Martin, L., Muller, B., Ortiz Suárez, P. J., Dupont, Y., Romary, L., Villemonte de la Clergerie, E., Seddah, D., and ´ Sagot, B. (2019). CamemBERT: a Tasty French Language Model. *arXiv e-prints, page arXiv:1911.03894, Nov.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *In Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2,* NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Nozza, D., Bianchi, F., Hovy, D. (2020). What the [MASK]? Making sense of language-specific BERT models

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *In Proceedings of NAACLHLT*, pages 2227–2237.

Pires, T., Schlinger, E. and Garrette, D. (2019). How multilingual is multilingual bert? *arXiv:1906.01502.*

Polignano, M., Basile, P., de Gemmis, M., Semeraro, G., and Basile, V. (2019). AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets*. In Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019),* volume 2481. CEUR.

Radford, A., Narasimhan K., Salimans, T., Sutskever, I., (2018) Improving Language Understanding by Generative Pre-training. *Technical report, OpenAI*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.* doi: 10.18653/v1/D16-1264. URL https: //www.aclweb.org/anthology/D16-1264.

Turing, A. (1950) Computing Machinery and Intelligence. *Mind, LIX (236):* 433–460

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems,* pages 5998–6008.

Virtanen, A., Kanerva, J., Ilo, R., Luoma, J., Luotolahti, J., Salakoski, T., Ginter, F., and Pyysalo, S. (2019). Multilingual is not enough: Bert for finnish. *arXiv preprint arXiv:1912.07076.*

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Proceedings of the 2018 EMNLP Workshop BlackboxNLP: *Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November. Association for Computational Linguistics.

Wu, Y., Schuster, M., Chen, Z., Quoc V Le, Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K. et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144.*