university of groningen / faculty of science and engineering

Probing Pre-trained Large Language Models
for Narrative Coherence

Master's Research Project
Roy David (S2764989)
August 1, 2023

Internal Supervisor(s): dr. S.M. (Stephen) Jones (Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen)
External Supervisor: dr. T. (Tommaso) Caselli (Computational Linguistics, University of Groningen)

Computational Cognitive Science
University of Groningen, The Netherlands

ABSTRACT

Probing tasks can be used to explore the capabilities of large language models (LLMs) in terms of their ability to encode linguistic knowledge and how they process (coherent) sequences of text, by using the models' representations to solve a task (proxied by a dataset). Transformer-based LLMs, such as BERT, have shown to be able to encode linguistic knowledge and dominate the state-of-the-art in a variety of NLP tasks. The extend to which these pre-trained large language models (PTLLMs) capture narrative coherence, given (coherent) sequences of text and a set of possible ending/follow-up sequences, in a zero-shot, multilingual setting has not been explored yet. This research presents an extensive study of the abilities of six PTLLMs, two multi-lingual (mDeBERTaV3 and XML-RoBERTa) and four monolingual language models (English: BERT, RoBERTa; Dutch: BERTje, RobBERTV2), to encode narrative coherence across sixteen datasets, consisting of either: short fictional stories or short news article narratives, with each several alternative variations, with varying narrativity types and coherence complexity. In addition we introduce a (small) language specific dataset for Dutch.

Our results show that these PTLLMs can capture narrative coherence mostly when having access to the full text and in simple cases, namely when the possible follow-up sequences do not present subtle linguistic differences and do not require complex commonsense reasoning. In most of these instances, the higher layers (8-12) yield the best performance. Moreover, when the data presented consists of short, coherent sentences with subtle linguistic differences between possible ending-sequences, the models' performance tends to drop (≈0.2 points) compared to the simple(r) cases, however still capturing (some) coherence. However, the models fail to capture coherence when the data presented consists of long(er) format sentences and subtle linguistic differences are present between the possible follow-up sequences. At the same time, simple probes show competitive results when compared to state-of-the-art systems on the same task and outperform all our baselines.

Keywords: Probing; Pre-trained Large Language Models; Natural Language Processing; Natural Language Understanding; Narrative Understanding; Narrative Coherence; Transformers; Cloze Task; Multilingual; Zero-shot; Contextualized Embeddings.

CONTENTS

PREFACE

After 8 years of full- and part-time studying, I can say that I thoroughly enjoyed my time as a student. I would like to thank all staff and fellow students who have helped me during my time at the University of Groningen. In particular, I would like to thank dr. S.M. (Stephen) Jones and dr. T. (Tommaso) Caselli for their help and guidance during my graduation project.

1 INTRODUCTION

Narrative Understanding (NU) is a high-level cognitive ability requiring the disentanglement and identification of multiple linguistic features based on the internal coherence of logically and temporally connected sequences in a story, which can have different degrees of narrativity (Abbott, 2014). Narrative coherence aims to assess the degree to which a story makes sense (Fisher, 1984, 1985), where previous work on narrative coherence focused on the temporal unfolding of event sequences and the modeling of logical causal relations (Chambers and Jurafsky, 2009; UzZaman et al., 2013; Minard et al., 2015; Granroth-Wilding and Clark, 2016; Mirza and Tonelli, 2016; Mostafazadeh et al., 2016; Caselli and Vossen, 2017; Weber et al., 2018), with few works testing broader notions of narrative coherence involving storytelling and commonsense reasoning between some input sequences and a set of possible target sequences (Mostafazadeh et al., 2017; Sharma et al., 2018a; Angelidis et al., 2019; Lal et al., 2021).

Transformer-based (Vaswani et al., 2017), using only its encoder, pre-trained large language models (PTLLMs), also known as auto-encoders, such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) and XLM-RoBERTa (XLM-R) (Conneau et al., 2019) have shown to be the state-of-the-art in most Natural Language Processing (NLP) tasks, surpassing Recurrent Neural Networks (RNNs) (Rumelhart et al., 1985; Hochreiter and Schmidhuber, 1997; Cho et al., 2014; Huang et al., 2015) and static word-embeddings (such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), fastText (Bojanowski et al., 2017)). These transformer-based PTLLMs are able encode more linguistic information in their representations compared to the earlier static pre-trained language models, as they are able to retrieve contextualized word-embedding representations, due to the attention mechanism of the transformer architecture and novel pre-training tasks.

The adoption of these PTLLMs (Howard and Ruder, 2018; Devlin et al., 2019; Radford et al., 2019) and increasing research using these contextualized representations on several downstream natural language processing (NLP) tasks to increase performance, invoked the interest and need for research focusing on the abilities of PTLLMs to encode linguistic properties. Research investigating PTLLMs on their abilities to encode linguistic properties is known as probing. Within the probing framework, **probing tasks** function as ways to explore what linguistic knowledge is possibly encoded in PTLLMs/(deep) neural models (Conneau et al., 2018), assuming that PTLLMs are large repositories of linguistic information (Derby et al., 2021; Mosbach et al., 2020; Miaschi et al., 2020), by using the models' representations to solve a task (proxied by a dataset). The main motivation behind these probing tasks is the opaqueness of the PTLLMs' representations (what and how much linguistic information they encode/capture). Such probing consists of a probing classifier (simple (linear models) or complex (multilayer perceptron etc.), probing task (i.e. specific linguistic property), probing dataset (which is used to probe the probing classsifier for the probing task), controls/evaluation-metrics (what and how to evaluate the output of the probing classifier). Although the idea behind this probing framework seems quite straightforward - training a classifier to predict some linguistic property using a models' representations given some task/dataset - the choice of task/dataset, classifier, evaluation metrics and interpretation of the results present each their own set of possible requirements and limitations (Belinkov, 2022).[1] Such probing research has shown that BERT is able to represent the steps

[1] For a critical discussion on probing see Belinkov (2022).

of the traditional NLP pipeline (POS tagging, parsing, NER, semantic roles, coreference) (Tenney et al., 2019a), where recent discourse-level probing tasks have focused on temporal processing (Vashishtha et al., 2020; Caselli et al., 2022), discourse structure (Kurfalı and Östling, 2021; Koto et al., 2021a), the information status of the entities (Loáiciga et al., 2022), anaphoric relations (Sorodoc et al., 2020; Pandit and Hou, 2021), discourse connectives and implicatures (Pandia et al., 2021), and script generation (Jin et al., 2022; Sancheti and Rudinger, 2022).

Whilst probing research on transfomer-based PTLLMs have been done on several NLP tasks (Tenney et al., 2019a; de Vries et al., 2020a, *inter alia*), probing multiple PTLLMs for their ability to encode narrative coherence in a multilingual, zero-shot setting has not been explored yet. To achieve this, specific cloze task datasets can be presented - to a probing classifier using the PTLLMs' representations - as a device to probe for narrative coherence. Cloze tasks focusing on NU, such as the Story Cloze Test (SCT) (Mostafazadeh et al., 2017) and the ((Coherent) Multiple Choice) Narrative Cloze (((C)MC)NC) (Chambers and Jurafsky, 2008; Granroth-Wilding and Clark, 2016; Weber et al., 2018), can be used to focus on exploring a models' capabilities of linguistic knowledge. Such cloze tasks require a model to choose the next best-fitting text-sequence from a set of possible follow-up sequences, given some context. Naturally, a key property of understanding such narratives and identifying the next best-fitting text-sequence is understanding/assessing its internal consistency, i.e., narrative coherence. Furthermore, cloze task datasets are sparse for languages other than English.

Given the ability of transformer-based PTLLMs to encode contextual representations, the lack of probing research testing broader notions of narrative coherence and the availability of cloze task type datasets to possibly probe for a key property in NU namely, narrative coherence, we can wonder if these contextual representations contain enough linguistic information to keep track of some form of coherence when processing temporally and logically connected sequences. Or put more simple, do contextualized representations encode linguistic information related to narrative coherence?

This research focuses on the abilities of PTLLMs to encode narrative coherence. The main **research question** for this research is: To what extent do PTLLMs encode narrative coherence to address the identification of temporally and logically connected sequences? This will be done using per layer simple probes (Tenney et al., 2019b; Vulić et al., 2020; de Vries et al., 2020b; Caselli et al., 2022, *inter alia*) on several different cloze task datasets. Our **contributions** can be summarised as follows: (i) we explore pre-trained (large) language models for narrative coherence, using single-layer simple probes, in a zero-shot, multi-lingual setting on sixteen cloze task datasets, with varying degrees of narrativity (Abbott, 2014) and coherence complexity in English and Dutch; (ii) we study the impact of multiple input representations, ranging from full text to event triggers; (iii) we probe and evaluate six PTLLMs, two multi-lingual (`mDeBERTaV3` (He et al., 2021) and `XLM-RoBERTa` (Conneau et al., 2019)) and four mono-lingual models (English: `BERT` (Devlin et al., 2018) and `RoBERTa` (Liu et al., 2019), Dutch: `BERTje` (De Vries et al., 2019) and `RobBERTV2` (Delobelle et al., 2020)), differing in their pre-training objectives, number of parameters, and size of data used in pre-training; (iv) in addition we present a small Dutch cloze task dataset.[2]

[2] Code and data are publicly available: `https://github.com/roydavid957/MRP_CCS`.

2 BACKGROUND

2.1 LANGUAGE MODELS

Large language models (LLMs) develop a statistical understanding of the language/-data it has been trained on by training these models on large amounts of raw text data in a self-supervised way. However only this statistical understanding is not enough to perform well on specific NLP tasks. These generic pre-trained large language models (PTLLMs) are then fine-tuned in a supervised way given a specific (pre-training) objective/task/dataset. This process is also known as transfer-learning.

Vaswani et al. (2017) introduced the Transformer model, relying on an attention mechanism to draw global dependencies between input and output, allowing for more parallelization. They showed that their model reached a new state-of-the-art in the (machine) translation space. The Transformer model consists of two blocks: Encoder and Decoder. The Encoder block builds a representation of the input, the Decoder block uses the representations of the Encoder, together with some other inputs, to generate a target sequence. The Encoder is optimized for acquiring information from the input, whereas the Decoder is optimized for generating outputs (Figure 1).

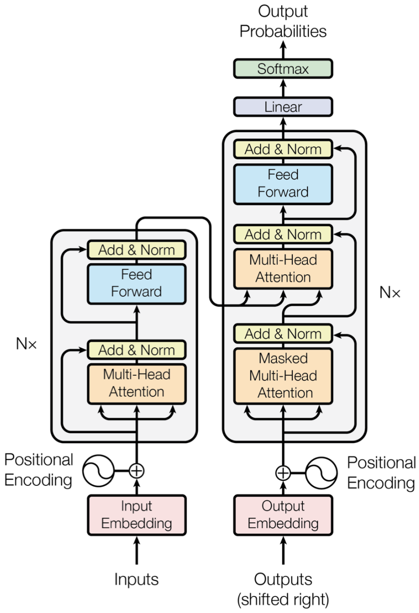


Figure 1: Transfomer - model architecture. On the left the Encoder, on the right the Decoder. Figure from Vaswani et al. (2017).

Transformer-based PTLLMs such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018), build upon (Vaswani et al., 2017), to build contextualized word embeddings, only using the Encoder model. They used masked language modeling (MLM) and Next Sentece Prediction (NSP) as pre-training objective, which allows for pre-training a deep bidirectional Transformer. They showed BERT to be state-of-the art at 11 NLP tasks. With the release of BERT came mBERT, a multilingual version of BERT pre-trained on 104 languages using Wikipedia. Following multi-lingual BERT (mBERT), Conneau et al. (2019) introduced XLM-RoBERTa (XLM-R), surpassing mBERT on a variety of cross-lingual benchmarks. They showed that the curse of multilinguality could be alleviated by increasing model capacity. These contextualized language models/PTLLMs are able to encode these contextualized representations, by leveraging the transformer's attention-mechanism, giving the model bidirectional access to the tokens in a sequence of text, and specific pre-training objectives such as MLM or Replaced Token Detection (RTD). These mechanics allow these language models/PTLLMs to build a contextualized representation of each token in a text-sequence by allowing bidirectional access to each token in a sentence, replacing neural architectures, such as RNNs (Rumelhart et al., 1985; Hochreiter and Schmidhuber, 1997; Cho et al., 2014; Huang et al., 2015), that relied on static word embeddings (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2017), where each word is assigned a single (type-level) vector.

2.2 PROBING

With the rise of deep neural network models for natural language processing (NLP) tasks, came the interest in interpreting and analyzing them. The main motivation behind this is the opaqueness of the models' representations (what and how much linguistic knowledge they capture). These models are analyzed by probing them for specific linguistic properties by solving a specific NLP task, proxied by a dataset. Such probing tasks can be used to explore the capabilities of language models in terms of their ability to encode linguistic properties, using the models' representations, to solve a task (proxied by a dataset). Belinkov (2022) provided a critical discussion on probing and showed that although the idea behind probing seems quite straightforward, choice of task/dataset, classifier, evaluation metrics and interpretation of the results are all key elements to take into account when designing and executing probing research. Moreover, they conclude their work with some key notions to keep in mind when designing probing experiments: (i) clearly define the original task(s), dataset(s), model(s) and the probing task(s), dataset(s), classifier(s); (ii) set upper and lower performance bounds and proper controls: control tasks (for word-level properties), datasets (for sentence-level properties); (iii) measure the probe's complexity (if ease of extractability is in question), the accuracy–complexity trade-off (when designing new probes), perform an intervention (to measure usage of information by the original model); (iv) using simple probes may avoid some of the issues about what the probe learns, compared to complex probes (i.e. linear classifier using the (zero-shot) representations from the internal model *vs.* a fine-tuned PTLLM using a multilayer perceptron).

Previous, pre-transformer, probing work focused mainly on probing using a neural models' hidden states and static embeddings, where some of the first probing studies focused on providing more nuanced evaluations of word embeddings by training classifiers on static word embeddings to predict various NLP properties, rather than integrating them in downstream tasks (Köhn, 2015; Gupta et al., 2015).[1] Recent work on probing contextualized PTLLMs on different NLP tasks, included work on versions of BERT and XLM-R. Tenney et al. (2019a) and de Vries et al. (2020a) showed that the pipeline-like behaviour is present in both a monolingual pre-trained BERT-based model as well as a multilingual model even though task-specific information is distributed between layers. Caselli et al. (2022) used XLM-R to investigate how PTLLMs encode information about events and their temporal ordering in a multilingual setting. They compared the default settings with: (i) only the embeddings of the events in the pair; (ii) the embeddings from two XLM-R base models previously fine-tuned with the EN-TimeBank and EN-TB-Dense corpora; (iii) monolingual static word embeddings. Their probing results indicate

[1] For a survey on the probing framework up to 2019 see Belinkov and Glass (2019).

that adding more information to lexical entities is detrimental. Furthermore they showed that temporal relation classification between events can be a tough task for PTMLs. Both papers used single layer probes to train a linear Support Vector Machine (SVM) (Boser et al., 1992).

Wilner et al. (2021) applied a transformer model on narrative event representations. Here, a narrative can be seen as a series of events, where an event consists of a predicate and its relevant roles (Chambers and Jurafsky, 2008; Granroth-Wilding and Clark, 2016). Using chains of events that are expected to be in the same story, a model can be taught to encode something like a script (Wilner et al., 2021). Thus to understand narrative representations, a system is required to have temporal and logical knowledge of connected sequences of events. They showed that using attention to re-contextualize events across the whole story achieves state-of-the-art performance on the Multiple Choice Narrative Cloze (Granroth-Wilding and Clark, 2016) and scoring competitively on the Story Cloze Task (Mostafazadeh et al., 2017; Sharma et al., 2018a). Furthermore they found that verbs carry event semantics in BERT.

2.3 CLOZE TASKS

Cloze tasks are usually "fill in the blank" tasks, such as the CLM and MLM pretraining objective. These task require systems to fill in some blank spot by generating their own output or choosing from a set of possible options. Such tasks enable models to show their NLP/U capabilities and requires some linguistic knowledge when being proxied by such a task. By selecting, modifying and/or creating text-based datasets to fit a specific cloze task, we can use them to probe (language) models for specific linguistic knowledge. In this way cloze task datasets can be presented as a device to probe language models for specific linguistic properties.

In line with Chambers and Jurafsky (2008), after the work of Rudinger et al. (2015) and inspired by Sadeghi (2014), Granroth-Wilding and Clark (2016) proposed a multiple choice version of the Narrative Cloze Task (NCT), Multiple Choice Narrative Cloze (MCNC). In this task, a system is presented with a series of contextual events and needs to choose the next, observed, event out of a set of multiple possible events. This procedure allows to compare models that take account of richer information from the text about both context and candidate events. Building upon Granroth-Wilding and Clark (2016), Weber et al. (2018) proposed a (filtered) version of the MCNC, the Coherent Multiple Choice Narrative Cloze (CMCNC), that accounts for frequency cutoffs of common events and improves coherency across the narratives.

Similarly, but more story driven, is the LSDSem'17 shared task (Mostafazadeh et al., 2017). The shared task included the Story Cloze Test, for the evaluation of story understanding and script learning of systems. It involved a system to choose the right ending to a given four-sentence story, out of two plausible story ending sentences. The task was to determine which out of the two sentences is most plausible given a short story as context. The cloze task was created using the ROCStories dataset[2]. Following this Sharma et al. (2018a) shed some light on the human-authorship biases discovered in the SCT (`SCTv1.0`) dataset. They created the `SCT-v1.5` dataset, to overcome some of the biases.

2.4 KEY FINDINGS

Vaswani et al. (2017), Devlin et al. (2018) and Conneau et al. (2019) are the foundation of the model architecture and the PTLLMs/contextualized word embedding

[2] https://www.cs.rochester.edu/nlp/rocstories/

models that will be used and probed in this research. Tenney et al. (2019a), de Vries et al. (2020a) and Caselli et al. (2022) provide this research with a roadmap on how to probe PTLLMs, using single layer probes and a linear classifier. Moreover, Wilner et al. (2021) showed that transformer models can be used for narrative representations. Chambers and Jurafsky (2008); Granroth-Wilding and Clark (2016); Weber et al. (2018) and Mostafazadeh et al. (2017); Sharma et al. (2018a) provide the source of the data and objective of this research. Lastly, Belinkov (2022) provides us with some key notions to keep in mind on the complexity, limitations and use of the probing framework.

3 DATA AND MATERIAL

In this chapter we will discuss and describe the datasets used for this research as well as any preprocessing that has been done.

We have selected, created and used subsets of several (variations of) datasets, varying in topic, complexity and language, resulting in sixteen datasets in total, in both English and Dutch, where models have access to an initial context to make decisions on what is the best ending or follow-up from a set of two possibilities. The ending alternatives and their complexity offer variations in the expression of narrative coherence. The datasets differ from each other in: (i) type: short fictional stories, news-article narratives; (ii) complexity: short(er) format sentences (short (fictional) sentences), long(er) format sentences (news-article style); (iii) incorrect final sentence alternative (relative to the input context): random, coherent, same document; (iv) narrativity (Abbott, 2014): short fictional stories with story-ending final sentence and mostly one protagonist, short news-article narrative text excerpts with a follow-up final sentence and (possibly) multiple protagonists; (v) language: English and Dutch.

As cloze tasks type datasets are sparse/non-existent for Dutch and since we aim to explore PTLLMs in a multi-lingual setting, we used an open-source machine-translation (MT) model to translate the English datasets (and their variations) to Dutch. The machine-translated (MT) Dutch variations of the English datasets were created using an open-source English to Dutch machine-translation model (Tiedemann and Thottingal, 2020)[1]. They provide open translation services and tools that are free from commercial interests and restrictions. Their models are based on transformer-based neural machine translation (NMT), Marian-NMT (Junczys-Dowmunt et al., 2018). Their models are trained on open-source parallel corpora collected in the large bitext repository OPUS (Tiedemann, 2012). The architecture is based on a standard transformer setup with 6 self-attentive layers, in both the encoder and decoder network, with 8 attention heads in each layer.

For all datasets the event triggers have been identified by selecting the verbs with `ROOT` label from the SpaCy dependency parsing. If the `ROOT` is not assigned to a verb, then we select the first verb in the sentence. In case no verb is available, due to parsing errors, we used the sentence token classified as `ROOT`.

All datasets are in the same format: a 4-sentence story as context, with two possible ending options, one correct and one incorrect.

3.1 STORY CLOZE

The Story Cloze Test v1.0 (`SCT-v1.0`) (Mostafazadeh et al., 2017) and v1.5 (`SCT-v1.5`) (Sharma et al., 2018b) are composed of short fictional stories elicited from crowd workers, with a high degree of narrativity (Abbott, 2014). The benchmarks have a common structure where the objective is to pick the best story-ending sentence from two possible options, given a 4-sentence story context. The datasets require access to some form of commonsense knowledge, reasoning and understanding of storytelling, in order to make the correct decision for every story (Chaturvedi et al., 2017; Liu et al., 2018). These benchmarks have been designed to evaluate systems' NU abilities to identify the correct coherent story-ending sentence by relying entirely on the information of the preceding 4-sentence context, requiring access to

[1] https://huggingface.co/Helsinki-NLP/opus-mt-en-nl

some form of narrative coherence. Moreover, each sentence in the fictional stories has on average ≈9 words.

`SCT-v1.0` has known stylistic artifacts (e.g., differences in word-token count, sentiment, and sentence complexity between the right and the wrong ending) that influence the performance of systems but it has been previously used for discourse probing (Koto et al., 2021a). `SCT-v1.5` addresses the stylistic bias of `SCT-v1.0`. Unfortunately, we could not get access to the labels of the test set.[2] We thus decided to use the official validation data distribution in a 5-fold cross-validation experiment setting. The more challenging nature of `SCT-v1.5` will offer more realistic results on the narrative coherence abilities of PTLLMs.

Table 1 shows the data distribution for `SCT-v1.0` and `SCT-v1.5`. In all of our experiments, the validation split has been used for training (or cross-validating) the probing classifiers. Example (1) shows examples of data instances of the fictional stories from both `SCTv1.0` and `SCTv1.5` versions of the SCT, where the first four sentences function as input ((i) – (iv)), with two possible story-ending final sentences, with one correct (*True*) final sentence and one incorrect (*False*), but still coherent with the story, final sentence.

(1) `SCTv1.0`:
(i) Bindu planned a party with her friends.
(ii) They met at her house to discuss what food and band to use.
(iii) One of Bindu's friends brought samosas and doogh.
(iv) Four friends played music at the party.
True: Everyone had a great time.
False: Bindu hates her friends and parties.

`SCTv1.5`:
(i) Mary wanted to make plans for New Year's Eve.
(ii) She decided to have a party at her apartment.
(iii) She invited all her friends.
(iv) Her friends brought food and drinks to the party.
True: It was the best New Year's Eve party ever.
False: She accepted the food, and asked them all to leave.

Split	`SCT-v1.0`	`SCT-v1.5`
validation	1,871	1,571
test	1,871	–

Table 1: Data distribution of `SCT-v1.0` and `SCT-v1.5` used in our experiments. Figures refers to stories. Due to the unavailability of the golden labels for the test set of `SCTv1.5`, we opted for 5-fold cross-validation to evaluate the PTLLMs on this dataset.

3.2 NARRATIVE CLOZE

The original (Multiple Choice) Narrative Cloze Task ((MC)NCT) evaluated the abilities of systems to identify coherent sequences of events (Chambers and Jurafsky, 2008; Granroth-Wilding and Clark, 2016). Here, an event sequence was represented as a triplet of the form `SUBJ|VERB|OBJ` (e.g.,"*Gorbachev | surprised | leaders*"). Each event sequence had a common protagonist, either in subject or in object position.

For our probing experiments, we carved two different NCT datasets, derived from Granroth-Wilding and Clark (2016) and Weber et al. (2018), using the SCT

[2] We have contacted the organizers of the 2018 Story Cloze Task but we did not manage to get the evaluations of our predictions.

format (4-sentence context, two possible final sentence options (one correct, one incorrect)), by recovering the original sentence from the English Gigaword corpus for each event triplet. We further limited the data to narrative passages composed by a minimum of at least six sentences: four input sentences, one correct final sentence and (at least) one incorrect final sentence alternative option. Systems are challenged to decide on the correct next follow-up sentence between the held out target sentence and a random final sentence. The random sentences have been selected from the same dataset with the following criteria: they do not occur more than once and they belong to text excerpts other than the one in analysis. We call this dataset `NCT-Full`. The second dataset, `NCT-Human`, is a subset of `NCT-Full` where we select only text passages with a human protagonist (either in subject or object position). We used SpaCy[3] to identify the human protagonists. `NCT-Full` and `NCT-Human` have longer sentences than SCT, with an average of ≈26 words per sentence. As an alternative we created versions of these datasets with a same document *SameDoc* incorrect final sentence alternative, randomly selected from the same article text excerpt as the one in analysis, outside of the sentences already used for analysis (first four as input, fifth as correct final sentence). The range of the *SameDoc* alternatives, as a distance compared to the (last, fifth) correct final sentence, is between 1-181, with a mean of ≈12 (SD≈8) sentences.

There are two major differences between the NCT and SCT datasets: (i) NCT is based on text excerpts from news articles, having a lower narrativity than stories (Abbott, 2014); (ii) systems have to decide what is the next follow-up sentence, rather than the best story ending sentence.

Table 2 summarises the data distribution for the NCT datasets. Example (2) shows examples of data instances from both `Full` and `Human` versions of the NCT, where the first four sentences ((i) – (iv)) function as input, the fifth sentence in the narrative as the correct final sentence (*True*) with either a same document (*SameDoc*) or random (*Random*) incorrect final sentence alternative. `NCT-Full` shows a short narrative about a parking-lot at a graduation, `NCT-Human` shows a short narrative about an obituary for a woman named Sophie.

(2) `NCT-Full`:

(i) THE parking lot at my son's graduation is a sea of motorcycles: black, yellow, loud, smoke-belching, flame-adorned, sparkling with chrome so bright you have to look away.

(ii) Far from the dappled shade of any Ivy League campus, this blazing blacktop belongs to the Motorcycle Mechanics Institute in Phoenix, Ariz., a sprawling complex of freshly whitewashed, warehouse-size buildings with red and blue accent lines.

(iii) My husband steers our rented Nissan through the lot, searching for an empty spot among the motorcycles.

(iv) Hiding behind my sunglasses, I look around at the other parents and friends in their halter tops and jeans, scarf shirts, sleeveless T-shirts and turquoise bracelets.

True: And tattoos, of course, lots of them: roses, serpents, spiders, geometric patterns and sunbursts, explosions of red, blue and green.

SameDoc:My son – this young man I love so much but who has caused himself and his family such heartbreak over the past 20 years – is absolutely filled with joy.

Random:Each plays point guard because his team needs a point guard, but each is a true shooting guard not afraid to take the difficult shot.

`NCT-Human`:

(i) Age 90, of Forest Hills, New York, died peacefully November 30, 2004.

[3] https://spacy.io

(ii) Beloved wife of Sidney, loving mother of Susan, Daniel, daughter-in-law Adriane, adored grandmother of Michael Jake.

(iii) During WWII, Sophie worked for the Office of War Information and then went on to serve in the US Embassy in Moscow.

(iv) She lived her life with dignity, courage and great strength.

True: Sweet Sophie, your family will miss and remember you always.

SameDoc:Funeral services will be held 9:45 AM, on Thursday, December 2, at Gutterman's Funeral Home/Parkside Chapel, 98-60 Queens Boulevard, Rego Park, New York.

Random:For my generation of movie lovers (born after "A Streetcar Named Desire" and now stunned to be old enough to be Elijah Wood's father), being a Marlon Brando fan meant absorbing his work in reverse.

Split	NCT-Full	NCT-Human
train	3,159	1,470
test	4,757	2,148

Table 2: NCT datasets and splits used in our experiments. Figures refer to unique blocks of text passages (narratives); NCT-Full, NCT-Human are binary.

3.3 NARRATIVE CLOZE DUTCH

We created a Dutch version of the NCT-Full and NCT-Human datasets, called NCT-Full-Dutch, NCT-Human-Dutch, respectively or NCT-Dutch as a collective, using a subset of publicly available Dutch data provided by Yeh et al. (2019). This dataset was created for partisanship detection of Dutch news articles from DPG Media[4]. We added/created this cloze task dataset since (i) due to the long(er) format sentences of the NCT datasets, the open-source machine-translation models showed some artifacts in the MT-Dutch datasets and (ii) such cloze task datasets are sparse in languages other than English. To extract narratives following some common protagonist for the Dutch NCT datasets (NCT-Dutch), we used a multi-lingual coreference model and added this to the spaCy pipeline[5]. Since coreference models for languages besides Dutch are quite sparse/non-existent, we opted for cross-lingual coreference. This uses the assumption a trained model with English data and cross-lingual embeddings should work for other languages with a similar sentence structure. In this way we aimed to extract narratives that are similar to the English NCT dataset. This dataset is similar to NCT in that it has, on average, more words per sentence than the SCT datasets, however less than the NCT datasets, with ≈16 words per sentence. Same as for the NCT datasets, we created versions of these datasets with a same document (*SameDoc*) final sentence, randomly selected from the same article text excerpt as the one in analysis, outside of the first five sentences (first four used as input, fifth as correct final sentence). The range of the *SameDoc* alternatives, as a distance compared to the (last, fifth) correct final sentence, is between 1-86, with a mean of ≈12 (SD≈7) sentences.

Due to the small(er) size of these datasets, we opted for 5-fold cross-validation to evaluate the PTLLMs. Table 3 summarises the data distribution for the NCT-Dutch datasets. Example (3) shows examples of data instances from both Full and Human versions of the NCT-Dutch, where the first four sentences function as input, the fifth sentence in the narrative as the *True* or correct final sentence with either *SameDoc* or *Random* as the incorrect alternative final sentence. NCT-Full-Dutch shows a short

[4] https://www.dpgmediagroup.com/nl-NL

[5] https://spacy.io/universe/project/crosslingualcoreference

narrative about colon-cancer research, NCT-Human-Dutch shows a short narrative about a lawyer named Wevers.

(3) NCT-Full-Dutch:
- (i) Het bevolkingsonderzoek naar darmkanker is een groot succes.
- (ii) Dat blijkt vandaag uit onderzoek in opdracht van het RIVM.
- (iii) De poeptest onder senioren toont meer darmkanker aan dan voorzien en de opkomst is hoger dan verwacht.
- (iv) Als darmkanker vroeg wordt ontdekt, is de kans groter dat behandeling succes heeft.
- *True*: Door dit bevolkingsonderzoek verwacht het RIVM dat in de nabije toekomst jaarlijks 2250 sterfgevallen kunnen worden voorkomen.
- *SameDoc*:„Alhoewel elke complicatie er natuurlijk één te veel is."
- *Random*:Nergens op de wereld zijn mobiele netwerken zo goed en snel als in Nederland.

NCT-Human-Dutch:
- (i) Waarom het verdienmodel van jurist Wevers zo goed werkt We dachten, zegt vader Henk Wevers als ik binnenkom, 'dat het nu wel zo'n beetje voorbij zou zijn.'
- (ii) Maar het is niet voorbij.
- (iii) Kevin Wevers vond na zijn afstuderen als jurist geen baan, ging de bijstand in en moest als gemeentelijke tegenprestatie papieren bloemen vouwen in een fabriekshal - hij kreeg de participatiesamenleving recht in zijn gezicht.
- (iv) Begon in zijn ouderlijk huis een juridisch bureau, kocht een tweedehands Jaguar en werd in no-time held van ouderen en gehandicapten die vanwege de participatiesamenleving gekort worden op huishoudelijke hulp.
- *True*: En is dat tot zijn eigen verbazing nog steeds want de gemeenten, zegt hij, krijgen de boel niet op orde.
- *SameDoc*:De gemeenten vinden hem inmiddels een geldwolf, maar overschrijden zelf de juridische termijnen, 'en wij zijn meer dan coulant', zegt Kevin, 'dus wie heeft dan schuld?'
- *Random*:Mannen nemen vaker verlof op om bij de kinderen te zijn als de leidinggevende dat zelf ook gedaan heeft.

Split	**NCT-Full-Dutch**	**NCT-Human-Dutch**
train	856	314
test	–	–

Table 3: NCT-Dutch datasets and splits used in our experiments. Figures refer to unique blocks of text passages (narratives); NCT-Full-Dutch and NCT-Human-Dutch are all binary. Due to the small(er) size of the Dutch datasets, we opted for 5-fold cross-validation instead of a separate train and test set.

4 METHOD

In this chapter we will discuss and break-down the methods and analysis done for this research. We will discuss the PTLLMs used in this research and their differences, explain how we executed the single layer probing of the PTLLMs and discuss our evaluation metrics in detail, as well as the baselines used for comparison.

4.1 PRE-TRAINED LARGE LANGUAGE MODELS

For our cross-lingual experiments we chose two state-of-the-art PTLLMs with (somewhat) similar architectures: `mDeBERTaV3-base` and `XLM-RoBERTa-base`. `mDeBERTaV3` (He et al., 2021) is a multilingual PTLLM that improves on `BERT` and `RoBERTa` using a disentangled attention and enhanced mask decoder, using the Replaced Token Detection (RTD) pre-training objective (Clark et al., 2020). `mDeBERTaV3-base` uses Wikipedia and the BookCorpus as training materials. `XLM-R-base` (Conneau et al., 2019) is a multilingual version of `RoBERTa`, pre-trained on 2.5TB of filtered CommonCrawl data with 100 languages using the Masked Language Modeling (MLM) pre-training task. The two models have the same number of attention heads (12) but different sizes for the vocabularies and number of parameters.

For our monolingual experiments we needed monolingual PTLLMs that are (somewhat) comparable, based on architecture, to the cross-lingual PTLLMs, and available in both English and Dutch, for monolingual comparability. Based on these requirements we settled for `BERT-base` (Devlin et al., 2018), `RoBERTa-base` (Liu et al., 2019) for our monolingual English PTLLMs and `BERTje` (De Vries et al., 2019), `RobBERTV2` (Delobelle et al., 2020) for our monolingual Dutch PTLLMs, where `BERTje` is the Dutch equivalent of `BERT`, however trained on Dutch data without the NSP pre-training objective. Similarly, `RobBERT` is the Dutch equivalent of `RoBERTa`.

The main differences between all PTLLMs are: `BERT` was the first Transformer-based (only using the Encoder model) PTLLM, using the MLM and NSP pre-training objectives on English text. `BERTje` is the Dutch equivalent of `BERT`, but differs from `BERT` as it only used the MLM pre-training objective. `RoBERTa` improved upon `BERT` by increasing size of vocabulary and total number of trainable parameters and dropping the NSP pre-training objective. `RobBERT` is the Dutch equivalent of `RoBERTa` with lower vocabulary due to differences in raw text training data. `XLM-RoBERTa` is a multilingual version of `RoBERTa`, pre-trained on 2.5TB of filtered CommonCrawl data with 100 languages using the Masked Language Modeling (MLM) pre-training task. `mDeBERTa` is a multilingual version of `DeBERTa`, built upon `BERT` and `RoBERTa` with `ELECTRA` style pre-training: RTD, with half the model size of `XLM-RoBERTa` (based on vocabulary and total number of trainable parameters).

Table 4 shows a detailed overview of the PTLLMs in terms of number of attention heads, parameters, vocabulary size, total number of trainable parameters and pre-training objective(s).

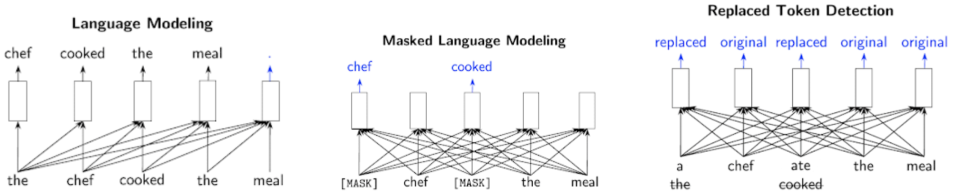
PRE-TRAINING OBJECTIVES Pre-training objectives are used after a language model has a statistical understanding of the raw text data is has been trained on. This is known as fine-tuning or transfer learning and is used to improve the generic pre-trained language models representations. Different pre-training objectives can be used for different tasks/goals/datasets.

On of the more traditional pre-training tasks is known as Causal Language Modeling (CLM). Here a system has to predict future tokens using the present and past

Model	L	H_m	H_{ff}	A	V	#params	Obj	huggingface.co/
XLM-RoBERTa	12	768	3072	12	250k	270M	MLM	xlm-roberta-base
mDeBERTaV3	12	768	3072	12	128K	98M	RTD	mdeberta-v3-base
RoBERTa	12	768	3072	12	50k	117M	MLM	roberta-base
BERT	12	768	3072	12	30k	110M	MLM+NSP	bert-base
BERTje	12	768	3072	12	30K	110M	MLM	GroNLP/bert-base-dutch-cased
RobBERTV2	12	768	3072	12	40K	117M	MLM	pdelobelle/robbert-v2-dutch-base

Table 4: Detailed per model overview. L: total number of layers; H_m: hidden size; H_{ff}: dimensions feed-forward layer; A: attention heads; V: vocabulary size (in tokens); #params: total number of trainable parameters; Obj: pre-training objective (MLM: Masked Language Modeling, RTD: Replaced Token Detection, NSP: Next Sentence Prediction).

tokens (see Figure 2a for a CLM example). Most transformer-based PTLLMs use the Masked Language Modeling (MLM) pre-training objective (Devlin et al., 2018), this masks specific words in a sentence, the model then has to predict these words based on the available words in the sentence, giving the need for bidirectional (past, present, future) of each token in a sentence. Creating contextualized embeddings as each token contains information about past and future tokens relative to the present token (see Figure 2b for a MLM example). As an improvement upon the MLM pre-training objective, Clark et al. (2020) introduced the Replaced Token Detection (RTD) pre-training objective. This pre-training objective consists of training two models: a generator and a discriminator, in a somewhat similar way as Generative Adversarial Networks (GANs) (Goodfellow et al., 2014). The generator is trained to predict and replace masked words (using a small MLM), then the discriminator has to identify these replaced words. After pre-training the generator gets dropped and the discriminator is kept. Recently, He et al. (2021) improved this ELECTRA-style pre-training by using gradient-disentangled embedding sharing for the generator and discriminator instead of sharing vanilla input word embeddings. They showed that this vanilla input word embedding sharing between generator and discriminator causes a tug-of-war dynamic as both models pull on these embeddings. This affects the training losses of both models on the token embeddings (see Figure 2c for a RTD example). Both the MLM and RTD pre-training objectives were introduced after the introduction of the Transformer model architecture (Vaswani et al., 2017).



(a) CLM

(b) MLM

(c) RTD

Figure 2: Overview of different pre-training objectives for language models, from left to right: Causal Language Modeling (CLM), Masked Language Modeling (MLM), Replaced Token Detection (RTD).[1]

Alternatively, BERT uses a combination of the MLM and Next Sentence Prediction (NSP) pre-training tasks. The NSP pre-training task consists of randomly selecting and concatenating two random sentences from the data, then the model has to predict if the sentences follow each other (binary task). Given the format of our probing task, where a model has to pick the next best-fitting sentence from a set of two possible options, given some context, BERT might have an advantage due to it being pre-trained in a partially similar fashion because of this NSP pre-training task.

[1] https://bit.ly/44U3g4M

4.2 NARRATIVE COHERENCE PROBING

Within the probing framework different probing tasks for different linguistic properties might need different probes in terms of complexity (Belinkov, 2022). One might use simple probes using the PTLLM's representations to train a linear probing classifier (Alain and Bengio, 2016; Hupkes et al., 2018; Liu et al., 2019; Maudslay et al., 2020, *inter alia*), alternatively more complex probes can be used, either by task-specific fine-tuning of the PTLLM's representations or using more complex probing classifiers, such as neural models (Conneau et al., 2018; Belinkov, 2018). However, the latter (might) cause some issues about what the probe learns (Belinkov, 2022). The simple probes use the (zero-shot) representations from the internal model to train a linear classifier, without needing to learn new parameters. This avoids some of the issues about what the probe learns. Since we are interested in analyzing the abilities of PTLLMs to encode narrative coherence in a zero-shot setting, we opt for simple probes.

Following previous work (Tenney et al., 2019b; Vulić et al., 2020; de Vries et al., 2020b; Caselli et al., 2022, *inter alia*), we extract embedding representations from each layer and use them to train a linear SVM whose objective is to predict the correct story ending for SCT or the correct follow-up sequence for NCT according to the preceding 4-sentence context. By default, we feed the input context to the SVM as concatenated embedding representations, each for every sentence in the context and for each option (separately). We then perform a binary classification task according to the dataset. Sentences are represented by averaging the embeddings of the tokens, excluding special tokens. We compare the default setting with three variations: (i) we merge all context sentences into a single text (*Full*), resulting in one embedding representation of the input context; (ii) we concatenate the embedding representations of the event triggers of each sentence context (*EvTr*); (iii) we extract the representations of the event triggers from the full context representation into a single vector, combining (i) and (ii) (*FullEv*). Example (1) shows examples of the settings used.

(1) *Default*: $[S_1]$+...+$[S_n]$
Full: $[S_1$, ..., $S_n]$
EvTr: [EVENT(S_1)+...+EVENT(S_n)]
FullEv: [EVENT(S_1), ..., EVENT(S_n)]

The probing classifier (linear SVM) was trained with the contextual embedding representations of these settings as input features, with either a 1 (true) or 0 (false) as a target, depending on whether or not the input feature contained the correct or incorrect final sentence (representations). The decision-making process of picking between the two possible final sentences was based on comparing the true probability of both the correct and incorrect instances. The possible final sentence alternative with the highest probability was predicted to be correct: label $= \arg\max(\{P_{\text{TRUE}}(y_1), P_{\text{TRUE}}(y_2)\})$ During the probing experiments we feed the context and one of the final sentence options to the PTLLM, given an input setting, the trained probing classifier then outputs the true probability based on the contextualized representations of the input (4-sentence context + final sentence option). After we have retrieved the true probabilities for both final sentence options, the final sentence option with the highest probability is predicted to be correct. Figure 3 shows the probing model architecture.

SUPPORT VECTOR MACHINE We used `scikit-learn` (Pedregosa et al., 2011) for the implementation of the linear Support Vector Classifier (SVC). The objective of a support vector machine (SVM) in binary classification is to find a hyperplane in an N-dimensional space, where N is the number of features, that classifies the

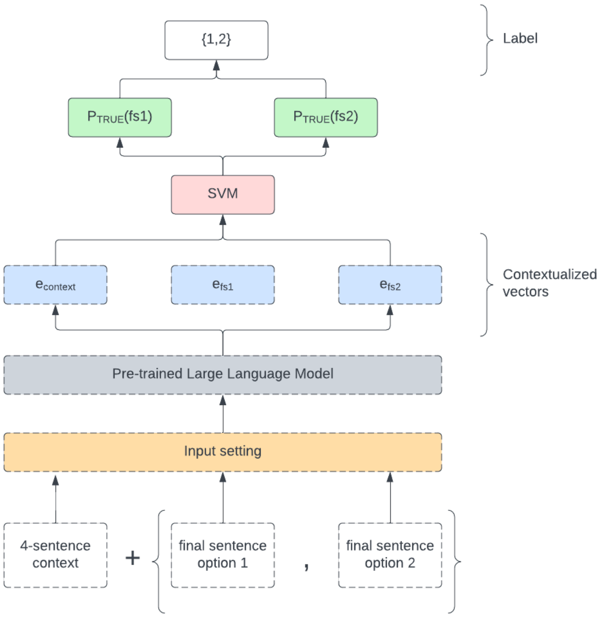


Figure 3: Probing model architecture. All parameters inside the dashed lines are fixed, while the SVM classifier uses the contextual vectors to output the true probabilities of both final sentence options, highest probability is predicted as the correct label.

data points, so that it maximizes the distance between data points of both classes, providing some reinforcement/margin for the classification of future data points.

4.3 EVALUATION

We evaluated the performance on the data using accuracy and compared all experiments to a random classifier (`random baseline`), a lexical classifier (`lexical baseline`) and a simple SVM classifier using a Term Frequency Inverse Document Frequency (TF-IDF) vectorizer (`tf-idf baseline`). Some datasets were evaluated using a fixed test set (`SCTv1.0`, all NCT datasets), others were evaluated using 5-fold cross-validation (`SCTv1.5`, all `NCT-Dutch` datasets).

The random baseline randomly picked between the two possible final sentence alternatives. The lexical baseline chose between the two possible story-ending or follow-up sentences based on lexical overlap between the 4-sentence context and the possible alternatives/targets. This overlap was determined by first removing stopwords and punctuation from the text. After that the remaining words were converted to their lemma. Then the lemmatized context was compared to the possible lemmatized story-ending/follow-up sequences. The possible alternative with the most overlap was chosen as the correct final sentence. Removing of the stopwords as well as the lemmatization of the remaining words were done using `spaCy`. Figure 4 shows the lexical baseline pipeline.

The TF-IDF baseline consisted of a SVM using a Term Frequency-Inverse Document Frequency (TF-IDF) vectorizer to turn the text into vectors. The tf-idf for a word is calculated by multiplying the term-frequency of a word in a document by the inverse-document-frequency of the word across a set of documents: tf-idf(t,d)=tf(t,d)×idf(t). The idf shows the rarity of a word in the entire document set and is computed as: $\text{idf}(t) = \log \frac{N}{1+\text{df}(t)}$, where N is the total number of documents and df(t) the number of documents that contain term t. The resulting tf-idf

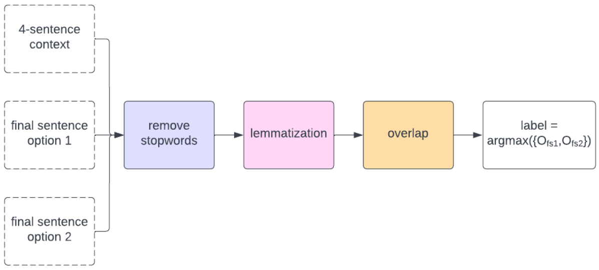


Figure 4: Lexical baseline pipeline. The 4-sentence context and each final sentence option goes through the pipeline of removing stopwords, lemmatizing the remaining words. After that the we count the overlap between the lemmatized words from the context and each final sentence option, the final sentence option with the most overlap is predicted to be the correct final sentence.

vectors are then normalized by the L2/Euclidean norm, by taking the square root of the sum of the squared vector values: $v_{\text{norm}} = \frac{v}{\sqrt{v_1^2 + v_2^2 + ... + v_n^2}}$. The text was fed to this model by merging/joining the 4-sentence context and the possible final sentence options individually (i.e. 4-sentence context+final sentence option 1 or final sentence option 2). The model was trained on these vectorized context input features with as target either a 1 if the input feature contained the correct final sentence option or 0 if the input feature contained the incorrect final sentence option. The decision-making process of picking between the correct/incorrect final sentence option was done in the same way as was done for the probing task. We used `scikit-learn` (Pedregosa et al., 2011) for the implementation of the TF-IDF baseline.

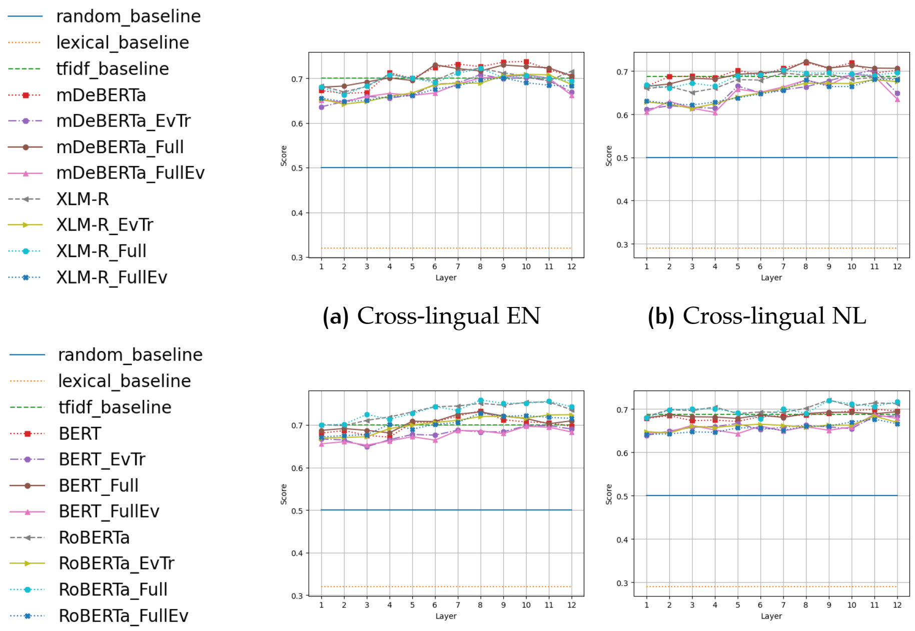
Lastly, where possible we compared the performance to the state-of-the-art on the same task. This was only possible for the `SCTv1.0` dataset.

5 RESULTS AND DISCUSSION

In this chapter we will give an overview of the scores per layer (1–12) for each model (multilingual: `mDeBERTaV3`, `XLM-RoBERTa`; monolingual English: `BERT`, `RoBERTa`; monolingual Dutch: `BERTje`, `RobBERTV2`) in the four experiment conditions (default, *Full*, *EvTr*, *FullEv*) per dataset (SCT: SCTv1.0, SCTv1.5; NCT: NCT-Full-Random, NCT-Human-Random, NCT-Full-SameDoc, NCT-Human-SameDoc; NCT-Dutch: NCT-Dutch-Full-Random, NCT-Dutch-Human-Random, NCT-Dutch-Full-SameDoc, NCT-Dutch-Human-SameDoc. Detailed per layer, per PTLLM, per dataset results are reported in Appendix 7.1.

5.1 STORY CLOZE

SCTV1.0 Figure 5 shows the per layer accuracy score for each model for the SCTv1.0 dataset. Figure 5a and Figure 5b show the per layer per cross-lingual model accuracy score results on the English and MT-Dutch variants of the dataset, respectively. Figure 5c and Figure 5d show the per layer per monolingual model accuracy score results. All figures show a similar trend in that the PTLLM's scores tend to (slowly) increase in performance, based on accuracy score, as the layers increase.



(a) Cross-lingual EN (b) Cross-lingual NL

(c) Monolingual EN (d) Monolingual NL

Figure 5: Per layer accuracy scores on the `SCT-v1.0` dataset. In the legends on the left: `XLM-R`, `mDeBERTa` (cross-lingual); `RoBERTa`, `BERT` (monolingual English); `RobBERT`, `BERTje` (monolingual Dutch), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

Furthermore, we can see that most models across all dataset variants outperform all baselines, across all layers, compared to the lexical and random baselines, or at least one/some layers, compared to the TF-IDF baseline, with the TF-IDF baseline being most competitive, compared to the best single-layer PTLLM scores. However as mentioned in Chapter 3, there is a known bias present within this dataset which might be the cause of the relative competitive performance of the TF-IDF baseline, compared to the PTLLMs' performance. For all models, the performance is lower in the earlier layers (1-4) of the model and tends to be the highest around the higher layers (8-12) of the model. Best performing model on the English datasets is the monolingual RoBERTa model with the full context as a single vector representation (*Full*) (0.759, Δ = 0.59 compared to the best performing baseline (tf-idf) = 0.7). The worst performing model on the English datasets is BERT with the *FullEv* setting, failing to outperform the tf-idf baseline (0.697, Δ = -0.003), based on best single layer score. Best performing model on the MT-Dutch datasets is mDeBERTa with *Full* settings (0.723), with RobBERT with *FullEv* settings as the worst performing model (0.677), based on best single layer score. The best performing layer per model were in the 8–11 layers, showing that the final layer does not result in best performance on the task.

Both cross-lingual figures (Figure 5a, 5b) show similar trends. The mDeBERTa models tend to (slightly) outperform the XLM-RoBERTa models on all experiment conditions apart from event triggers, with the default and *Full* context experiment conditions yielding the best performance for both models individually. Moreover, when the extracted event representations were used, both *FullEv* (extracted main events as one vector) and *EvTr* (extracted concatenated main event representations), these models tend to yield the worst performance, for both mDeBERTa and XLM-RoBERTa.

The monolingual figures (Figure 5c, 5d) show similar trends as well. Again the default and *Full* context settings appear to yield best performance, whilst the extracted event representations (*FullEv*, *EvTr*) yield the worst performance. Here the monolingual RoBERTa models (RoBERTa for English, RobBERTV2 for Dutch) tend to outperform the monolingual BERT models (BERT for English, BERTje for Dutch). For the English monolingual models the RoBERTa models tend to outperform the BERT models more clearly, whereas for the Dutch monolingual models the performance of the different PTLLMs tend to be a bit closer.

Comparing the performance on the English datasets to the MT-Dutch variants, we see a slight drop in performance. However this slight drop might be caused by machine-translation artifacts. The monolingual RoBERTa models tend to outperform the multilingual RoBERTa models, as expected. More notably is that the multilingual DeBERTa models tend to outperform some monolingual BERT models. These results show that multilingual models are able to be competitive with and in some cases outperform monolingual models.

Comparing our best performing PTLLM, based on single-layer accuracy, to the state-of-the-art on the same task, we see that our zero-shot PTLLM with the *Full* setting is able to outperform a comparable model using static pre-trained word-embeddings (Δ = 0.59) and score competitive with a fine-tuned ELECTRA model (Δ = -0.131).

SCTV1.5 Figure 7 shows the per layer accuracy score for each model for the SCTv1.5 dataset. Figure 7a and Figure 7b show the per layer per cross-lingual model accuracy score results on the English and Dutch variants of the dataset, respectively. Figure 7c and Figure 7d show the per layer per monolingual model accuracy score results. Most dataset and model variants, apart from monolingual Dutch, show a similar trend in that they tend to increase in performance, based on accuracy score, as the layers increase. However, the best performing layers per model range from the mid to high layers (6-10). Furthermore, we can see that all models across all dataset variants outperform the random and lexical baselines, across all layers (lexical- and random baseline) and for most models at least one/some layers compared to the TF-IDF baseline, with the TF-IDF baseline being most com-

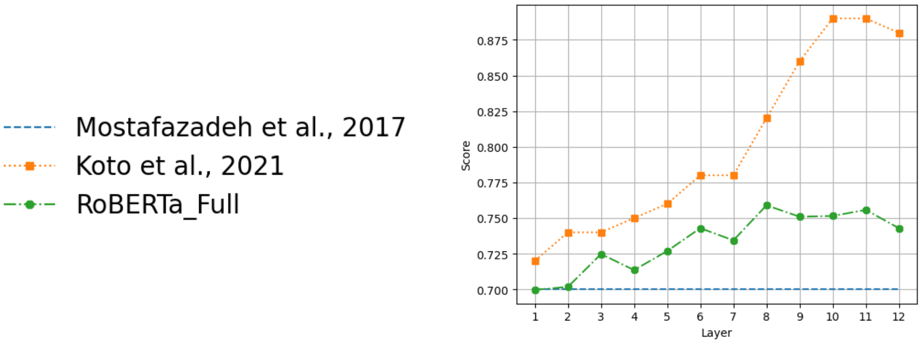
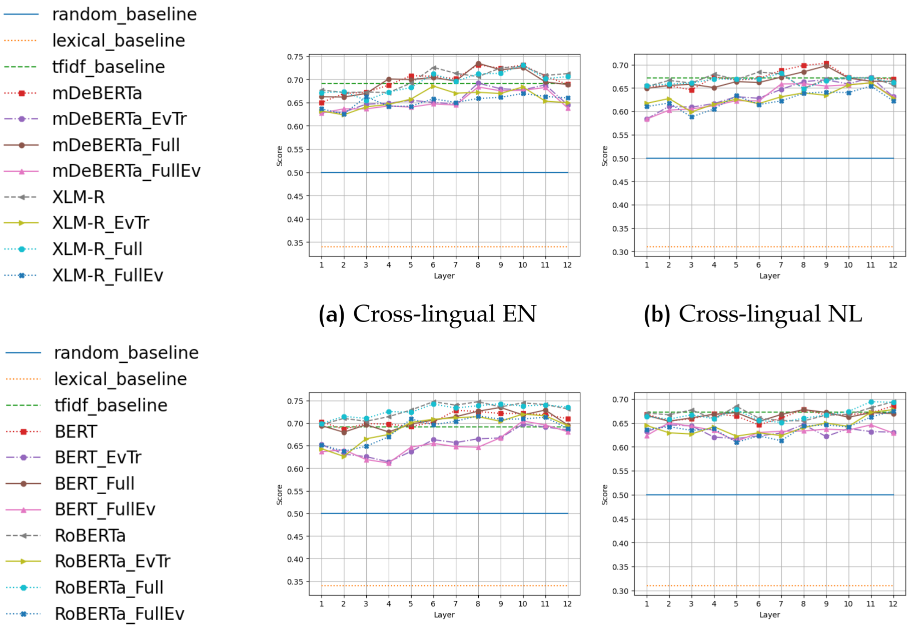


Figure 6: Per layer accuracy scores on the SCT-v1.0 dataset. Best model *vs.* state-of-the-art on same task, based on (single layer) accuracy. acoli used a comparable model, using SVM+static pre-trained word-embeddings (GloVe, word2vec) (Mostafazadeh et al., 2017), Koto et al. (2021b) used a fine-tuned ELECTRA model.



(a) Cross-lingual EN

(b) Cross-lingual NL

(c) Monolingual EN

(d) Monolingual NL

Figure 7: Per layer accuracy scores on the SCT-v1.5 dataset. In the legends on the left: XLM-R, mDeBERTa (cross-lingual); RoBERTa, BERT (monolingual English); RobBERT, BERTje (monolingual Dutch), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

petitive. RoBERTa with default settings (0.747) and mDeBERTa with default settings (0.703) yield the best performance on the English and MT-Dutch datasets, respectively. Whereas XLM-R with *FullEv* settings (0.669) and BERTje with *EvTr* settings (0.646) yield worst performance on the English and MT-Dutch datasets, respectively. Similar to the SCTv1.0 results, the extracted event representation settings (*FullEv*, *EvTr*) yield the worst results, whilst the default and *Full* settings yield the best results across all models and dataset variants. The multilingual models in the extracted event representation settings barely outperform (mDeBERTa with event triggers Δ = 0.01) or perform slightly worse than the TF-IDF baseline (XLM-R-EvTr Δ = -0.006, XLM-R-FullEv Δ = -0.022, mDeBERTa-FullEv Δ = -0.008). Interestingly, even though this dataset does account for the bias that was present in SCTv1.0, the TF-IDF baseline performs only slightly worse (Δ = -0.009) on this dataset compared to SCTv1.0. As expected, for three out of the four experiment conditions, the performance is lower in the earlier layers of the model (1–4) and tends to be the highest around the higher layers of the model (8–12). The only exception here being the monolingual Dutch models, where we see less of an upwards trend as the layers increase, where for some models, the performance even tends to stagnate or decrease as the layers increase (BERT-FullEv, -EvTr).

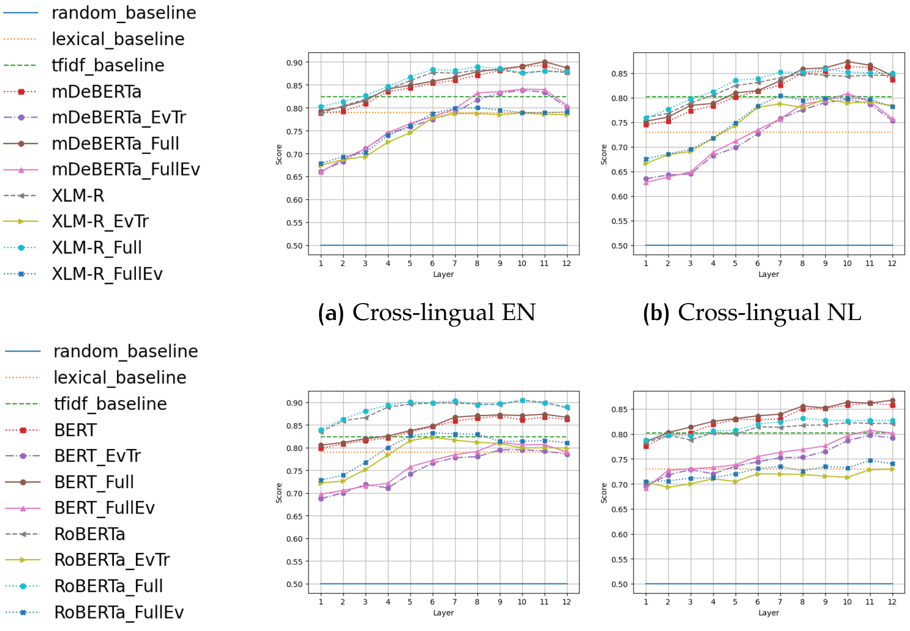
Both cross-lingual figures (Figure 7a, 7b) show similar trends. The mDeBERTa models tend to overall slightly outperform the XLM-RoBERTa models, with the default and *Full* settings yielding the best performance for both models individually. Moreover, when the extracted event representations were used, both *FullEv* and *EvTr*, these models tend to yield the worst performance, for both mDeBERTa (*FullEv*: 0.683, *EvTr*: 0.692) and XLM-RoBERTa (*FullEv*: 0.669, *EvTr*: 0.685).

The monolingual figures (Figure 7c, 7d show somewhat different trends. Again the default and *Full* settings appear to yield best performance, whilst the extracted event representations (*FullEv*, *EvTr*) yield the worst performance. Here the monolingual RoBERTa models (RoBERTa for English, RobBERTV2 for Dutch) tend to outperform the monolingual BERT models (BERT for English, BERTje for Dutch). For the English monolingual models the RoBERTa models tend to outperform the BERT models more clearly, whereas for the Dutch monolingual models the performance of the different PTLLMs tend to be a bit closer. However, the Dutch monolingual models seem to increase less as the layers increase, compared to the monolingual English models. Comparing cross-lingual Dutch with the monolingual Dutch models, we can see that the cross-lingual models appear to perform better or at least be very competitive/on par with the monolingual Dutch models. Showing that cross-lingual models can be very competitive and even outperform monolingual models in some instances.

Comparing the performance on the English datasets to the MT-Dutch datasets, we see a drop in overall performance. However this drop might be caused by machine-translation artifacts. The monolingual RoBERTa models tend to outperform the cross-lingual RoBERTa models, as expected. More notably is that the multilingual DeBERTa models tend to outperform the monolingual BERT models. Similar to SCTv1.0, these results show that multilingual models are able to be competitive with and in some cases outperform monolingual models.

5.2 NARRATIVE CLOZE

NCT-FULL-RANDOM Figure 8 shows the per layer accuracy score for each model for the NCT-Full-Random dataset. Figure 8a and Figure 8b show the per layer per cross-lingual model accuracy score results on the English and MT-Dutch variants of the dataset, respectively. Figure 8c and Figure 8d show the per layer per monolingual model accuracy score results. As already seen, all figures show a similar trend, as the layers increase the performance, based on accuracy score, tends to increase



(a) Cross-lingual EN **(b)** Cross-lingual NL

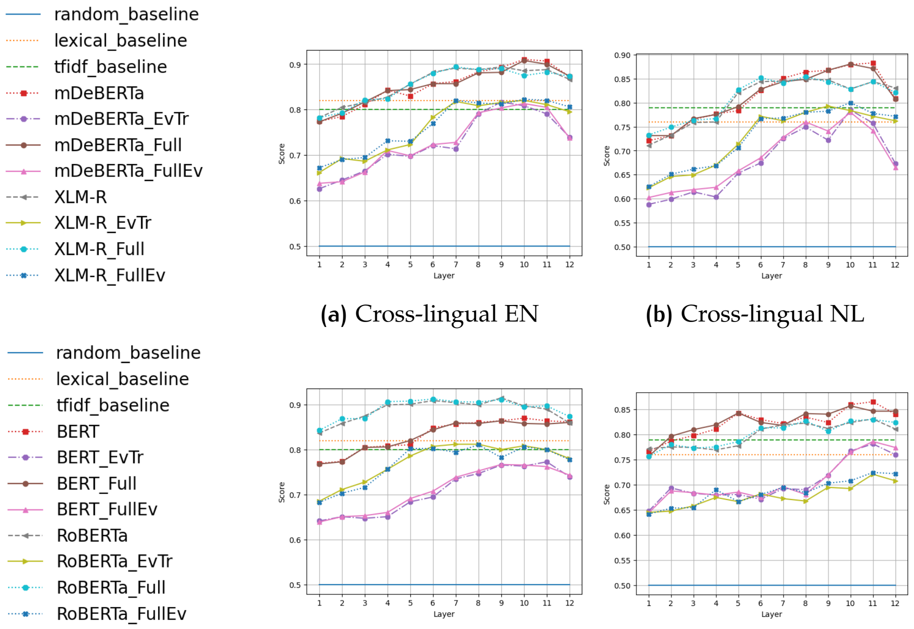
(c) Monolingual EN **(d)** Monolingual NL

Figure 8: Per layer accuracy scores on the `NCT-Full-Random` dataset. In the legends on the left: `XLM-R`, `mDeBERTa` (cross-lingual); `RoBERTa`, `BERT` (monolingual English); `RobBERT`, `BERTje` (monolingual Dutch), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

as well, with all models outperforming the random baseline (0.5) and most models outperforming both the lexical (0.79) and tf-idf baselines (0.824). The best performing layers per model range from the mid to higher layers (6–11), where the models on the English datasets have better performance compared to the MT-Dutch version. Similar to the Story Cloze results, this might be due to machine-translation artifacts, as artifacts might reduce the quality of the text and therefore possibly the coherence of the text passages. As expected, we can see that the models perform quite well on the random dataset where its easier to choose the correct option since the alternative has very minimal coherence or lexical overlap with the preceding context, as can be seen from the relatively high lexical baseline score (0.79). Best performing model on the English datasets is `RoBERTa` with default settings (0.905), with `XLM-R` with *EvTr* settings as the worst performing model (0.789), based on best single layer score. Best performing model on the MT-Dutch datasets is `mDeBERTa` with *Full* settings (0.873), with `XLM-R` with *EvTr* settings as the worst performing model (0.729), based on best single layer score. Interestingly we see that some models are outperformed by the TF-IDF baseline and are on par with/barely outperform the lexical baseline (`XLM-R-EvTr` = 0.795, `RoBERTa-EvTr` = 0.823, `BERT-EvTr` = 0.795, `XLM-R-FullEv` = 0.8, `BERT-FullEv` = 0.812), all of them in either the *FullEv* or *EvTr* setting.

`mDeBERTa` models outperform `XLM-R` models on all settings on both the English and MT-Dutch datasets apart from the event-trigger settings on the MT-Dutch dataset. Compared to the monolingual models, the multilingual models show competitive results, where on the English dataset `mDeBERTa` models outperform `BERT` on all settings, based on best single layer score, where on the MT-Dutch dataset `XLM-R` models outperform monolingual `RobBERT` models on all settings and `mDeBERTa` models on all settings apart from event-triggers. The monolingual `RoBERTa` models outperform the monolingual `BERT` models on all settings on the English dataset, whereas the `BERTje` models outperform `RobBERT` on all settings on the MT-Dutch dataset. Again, the *FullEv* and *EvTr* settings tend to yield the worst performance across both datasets, whereas the default and *Full* settings yield the best performance, based on best single layer score.

nct-human-random Figure 9 shows the per layer accuracy score for each model for the NCT-Human-Random datasets. Figure 9a and Figure 9b show the per layer per cross-lingual model accuracy score results on the English and MT-Dutch variants of the dataset, respectively. Figure 9c and Figure 9d show the per layer per monolingual model accuracy score results. In general we see similar trends here, compared to the NCT-Full-Random datasets. Higher layers (7–11) generally yield best performance. With the default and *Full* settings yielding best per model performance and *EvTr*, *FullEv* yielding worst per model performance, where for the MT-Dutch dataset all *EvTr* and *FullEv* models, apart from `XLM-R-EvTr` score lower than the tf-idf baseline (0.789), with `RobBERT-FullEv`, `-EvTr` scoring even lower than the lexical baseline (0.76, Δ = -0.032, -0.039, respectively). We see a similar trend in the English dataset results, where all *FullEv* and *EvTr* models, apart from `XLM-R-FullEv` fail to outperform the lexical baseline (0.82), with `BERT-EvTr`, `-FullEv` failing to outperform the tf-idf baseline as well (0.8, Δ = -0.007, -0.013). Similar to the NCT-Full-Random dataset the lexical and tf-idf baselines are relatively high, indicating less complexity within these random datasets. `RoBERTa` with default settings yields best overall performance on the English dataset (0.915) and `mDeBERTa` with default settings yields best overall performance on the MT-Dutch dataset (0.883), basde on best single layer score.



(a) Cross-lingual EN

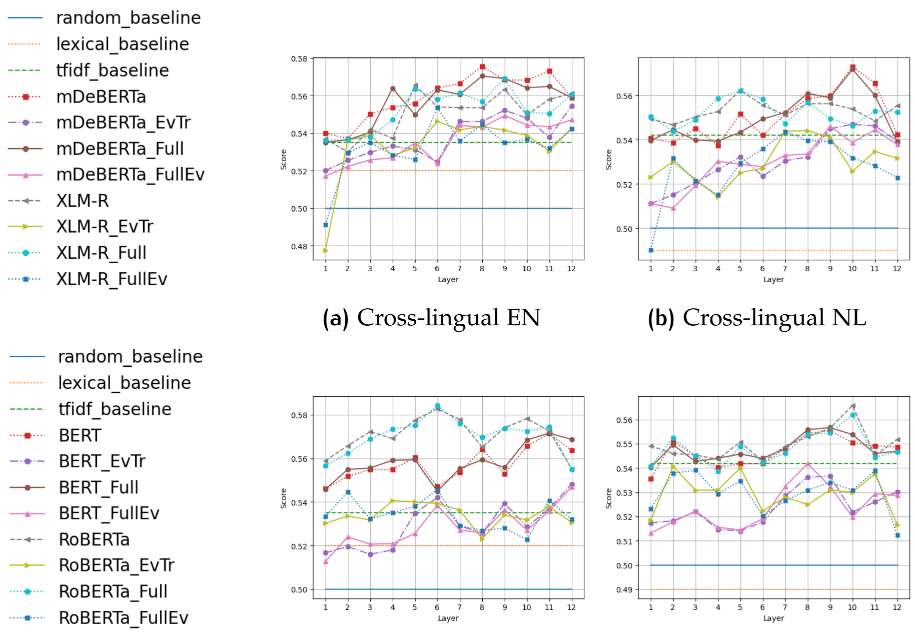
(b) Cross-lingual NL

(c) Monolingual EN

(d) Monolingual NL

Figure 9: Per layer accuracy scores on the `NCT-Human-Random` dataset. In the legends on the left: `XLM-R`, `mDeBERTa` (cross-lingual); `RoBERTa`, `BERT` (monolingual English); `RobBERT`, `BERTje` (monolingual Dutch), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

nct-full-samedoc Figure 10 shows the per layer accuracy score for each model for the NCT-Full-SameDoc dataset. Figure 10a and Figure 10b show the per layer per cross-lingual model accuracy score results on the English and MT-Dutch variants of the dataset, respectively. Figure 10c and Figure 10d show the per layer per monolingual model accuracy score results. Most models tend to outperform the



(a) Cross-lingual EN (b) Cross-lingual NL

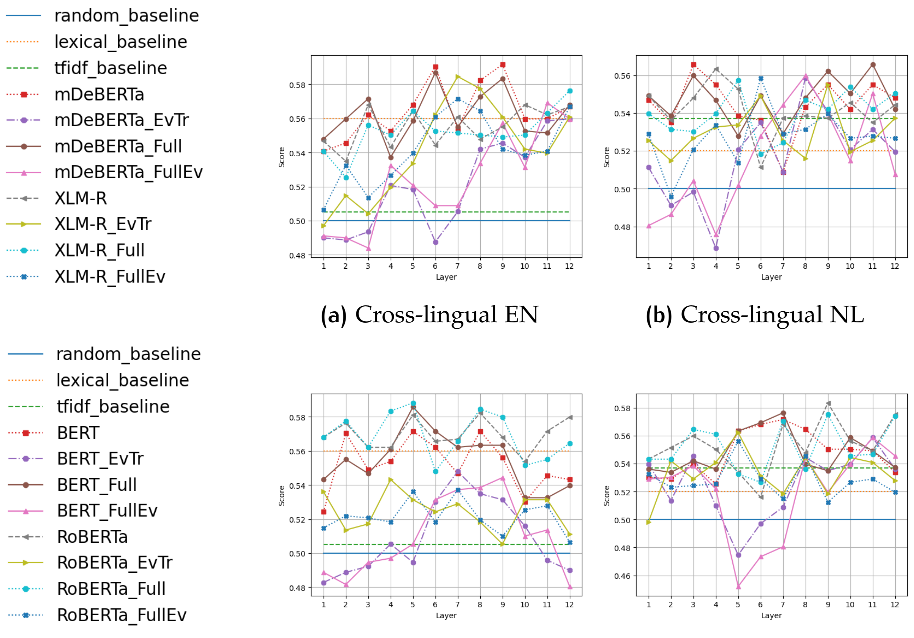
(c) Monolingual EN (d) Monolingual NL

Figure 10: Per layer accuracy score on the NCT-Full-SameDoc dataset. In the legends on the left: XLM-R, mDeBERTa (cross-lingual); RoBERTa, BERT (monolingual English); RobBERT, BERTje (monolingual Dutch), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

random (0.5) and lexical baselines (English: 0.52, MT-Dutch: 0.49) across all layers and settings or at least some/one layer compared to the tf-idf baseline (English: 0.535, MT-Dutch: 0.542), apart from RobBERT-FullEv, -EvTr (Δ = -0.003, -0.01) and BERTje-EvTr, -FullEv (Δ = -0.005, 0). Overall we see the models struggle more with consistency across the layers, yielding best layer performance as soon as layer 2 (RobBERT-EvTr) and layer 12 at the latest (mDeBERTa-EvTr, BERT-EvTr, -FullEv). Again the models show higher performance on the English datasets, compared to the MT-Dutch datasets, where the default and *Full* settings yield best per model performance. RoBERTa with *Full* setting yields best overall performance (0.584), showing that the monolingual model outperforms the multilingual model on the English dataset, whereas on the MT-Dutch dataset, multilingual DeBERTa with default settings yields best overall performance (0.573), based on best single layer performance. Worst performing model on the English dataset is RoBERTa with *EvTr* settings (0.541), with BERTje with *EvTr* settings being the worst performing model on the MT-Dutch dataset (0.537), based on best single layer score.

Overall we can see that the results, in absolute numbers, are (much) lower compared to NCT-Full-Random, making this a very challenging task. That this task is challenging is further indicated by the low(er) performance, small(er) differences between models, based on scores. The low performance and small differences, might also indicate that these results might be noise. We need statistical tests to verify what models are actually different from each other, however as we are only interested in the capabilities of PTLLMs to encode narrative coherence using different probing settings, this is beyond the scope of this research and leave this for future work. These results show that the PTLLMs fail to capture narrative coherence, due to the increased complexity of this dataset.

NCT-HUMAN-SAMEDOC Figure 11 shows the per layer accuracy score for each model for the NCT-Human-SameDoc dataset. Figure 11a and Figure 11b show the per layer per cross-lingual model accuracy score results on the English and MT-Dutch variants of the dataset, respectively. Figure 11c and Figure 11d show the per layer per monolingual model accuracy score results. Similar to the NCT-Full-SameDoc datasets we see a drop in performance compared to the random equivalent of this dataset. Showing less consistency across the layers, with best layer scores as early as layer 3 and as late as layer 12. All models and settings best performing layer outperform all baselines (random: 0.5, lexical: 0.52, tf-idf: 0.537) on the MT-Dutch datasets, whereas on the English dataset all models best performing layer outperform the random (0.5) and tf-idf (0.505) baselines, however we see that both the RoBERTa and BERT models with *FullEv* and *EvTr* settings fail to outperform the lexical baseline (RoBERTa-FullEv, -EvTr: Δ = -0.023, -0.017; BERT-FullEv, -EvTr: Δ = -0.016, -0.012). In general *FullEv* and *EvTr* settings yield the worst per model performance, whereas the default and *Full* settings yield best per model performance. With BERTje-Full and mDeBERTa with defaul settings yielding best overall performance (0.576, 0.592 respectively).



(a) Cross-lingual EN

(b) Cross-lingual NL

(c) Monolingual EN

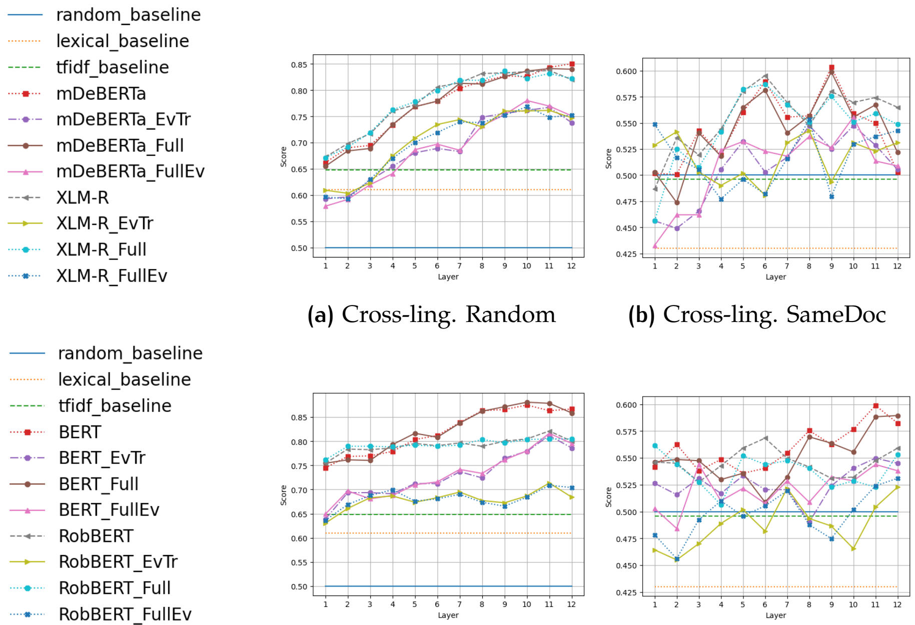
(d) Monolingual NL

Figure 11: Per layer accuracy scores on the NCT-Human-SameDoc dataset. In the legends on the left: XLM-R, mDeBERTa (cross-lingual); RoBERTa, BERT (monolingual English); RobBERT, BERTje (monolingual Dutch), for the default concatenated setting, Full for merged context sentences in a single text; EvTr for concatenation of event trigger representation only; FullEv for the merging of all event contexts into a single vector.

As already seen in NCT-Full-SameDoc, overall the results, in absolute numbers, are (much) lower compared to NCT-Human-Random, making this a very challenging task. That this task is challenging is, again, further indicated by the low(er) performance, small(er) differences between models, based on scores. The low performance and small differences indicate that these results might be noise. We need statistical tests to verify what models are actually different from each other, however as we are only interested in the capabilities of PTLLMs to encode narrative coherence using different probing settings, this is beyond the scope of this research and leave this for future work. These results show that the PTLLMs fail to capture narrative coherence, due to the increased complexity of this dataset.

5.3 NARRATIVE CLOZE DUTCH

NCT-DUTCH-FULL Figure 12 shows the per layer accuracy score for each model for the NCT-Dutch-Full dataset. Figure 12a and Figure 12b show the per layer per cross-lingual model accuracy score results on the Random and SameDoc (same article/narrative alternative follow-up sentence) variants of the dataset, respectively. Figure 12c and Figure 12d show the per layer per monolingual model accuracy score results on the Random and SameDoc variants of the dataset, respectively. Again we see a clear difference in performance between random and `SameDoc` version of the dataset. Not only in performance but also in trends, where in the random versions the peformance tends to increase as the layers increase, with best per layer performance in the higher layers (9–12). The `SameDoc` version show worse overall performance and less consistency across the layers, with per layer best models ranging from layers 6–12. All models outperform all baselines on both datasets (Random: `Random`: 0.5, lexical: 0.62, tf-idf: 0.751; `SameDoc`: random: 0.5, lexical: 0.43, tf-idf: 0.496). With default and *Full* settings yielding best per model performance and *FullEv*, *EvTr* yielding worst per model performance. With `BERTje-Full` and `mDeBERTa` with default settings yielding best overall performance on the random and `SameDoc` version of the dataset, (0.881, 0.604) respectively. These results indicate that the models are less capable of capturing coherence between context and possible follow-up sequences when the differences between the possible alternatives are linguistically more subtle, showing increased complexity.



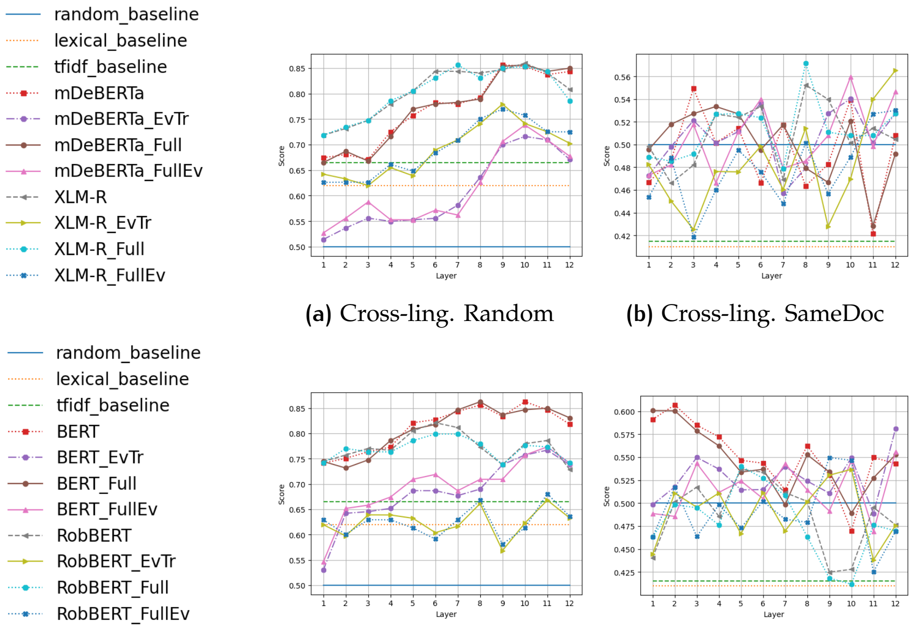
(a) Cross-ling. Random **(b)** Cross-ling. SameDoc

(c) Monoling. Random **(d)** Monoling. SameDoc

Figure 12: Per layer, 5-fold cross-validation, accuracy scores on the `NCT-Dutch-Full` dataset. In the legends on the left: `XLM-R`, `mDeBERTa` (cross-lingual); `RobBERT`, `BERTje` (monolingual), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

NCT-DUTCH-HUMAN Figure 13 shows the per layer accuracy score for each model for the NCT-Dutch-Human dataset. Figure 13a and Figure 13b show the per layer per cross-lingual model accuracy score results on the Random and SameDoc (same article/narrative alternative follow-up sentence) variants of the dataset, respectively.

Figure 13c and Figure 13d show the per layer per monolingual model accuracy score results on the Random and SameDoc variants of the dataset, respectively. We see similar trends compared to the NCT-Dutch-Full datasets. With the overall performance on the random dataset being higher compared to the SameDoc version. Furthermore showing the inconsistency in performance on the SameDoc version with best performing layers between layers 2–12, whilst the random version best performing layers are between layers 6–11. All models' best performing layer do outperform all baselines, both on random (random: 0.5, lexical: 0.62, tf-idf: 0.665) and SameDoc (random: 0.5, lexical: 0.41, tf-idf: 0.415). The default and *Full* settings yield best per model performance (SameDoc: `BERTje` with default settings (0.607), random: `BERTje-Full` (0.863)) and *FullEv*, *EvTr* yielding lowest per model performance (SameDoc: `XLM-R-FullEv`: 0.53; random: `RobBERT-EvTr`: 0.668). Again, showing that the contextualized extracted representations capture less linguistic information than the full text contextualized representations. Baselines scores on the SameDoc dataset are also lower than the random baseline, showing the added complexity of this dataset. Additionally, since this is a very small dataset, the tf-idf baseline might not have enough datapoints to train a well-fitted model. The PTLLMs are less effected by this, as we see less of a drop in performance compared to the drop in performance of the baselines (other than the random baseline), compared to the results on the NCT-Dutch-Full datasets.



(a) Cross-ling. Random

(b) Cross-ling. SameDoc

(c) Monoling. Random

(d) Monoling. SameDoc

Figure 13: Per layer, 5-fold cross-validation, accuracy scores on the `NCT-Dutch-Human` dataset. In the legends on the left: `XLM-R`, `mDeBERTa` (cross-lingual); `RobBERT`, `BERTje` (monolingual), for the default concatenated setting, *Full* for context sentences as a single text/representation; *EvTr* for concatenation of event trigger representations only; *FullEv* for the merging of all event contexts into a single representation.

5.4 DISCUSSION

Overall we see the same main trends across all models, datasets and experiment conditions. In general the default setting (input context as a per sentence concatenated embedding representations) or the input as the full context (*Full*: the input context as one embedding representation) yield the best performing models. This shows that the contextual embedding representations in these settings contain the most amount of useful information to complete the task. The event triggers (*EvTr*: per input context sentence extracted, concatenated main event embedding representation) and full context event triggers (*FullEv*: main event embedding representation extraction from the input context sentences as one embedding representation), showed to yield the worst performance. Indicating that these representations, although extracted from the context and therefore they should contain some contextual information about the sentence, given the attention mechanism of transformer models, contain less/not enough information to perform well on these tasks. However even the worst performing experiment conditions are able to consistently outperform most baselines, based on best single layer score, on most datasets. This shows that these embedding representations contain at least more contextual information than simple baselines in most of our experiments.

As the English `BERT` model has been pre-trained using the MLM and NSP objectives, we might expect better performance on these probing experiments, as the objective somewhat resembles the NSP pre-training objective. To reiterate, in this NSP pre-training objective two sentences were concatenated and the (`BERT`) model had to decide if the two sentences follow each other or not (binary task). For our probing experiments, the probing classifier was trained to predict if a final sentence fitted a 4-sentence input story/narrative (also binary task), using the PTLLM's representations of the text. As only the monolingual English `BERT` model was trained using this NSP pre-training objective we can only compare its results on the English datasets (6 in total: `SCTv1.0-EN`, `SCTv1.5-EN`, `NCT-Full-Random-EN`, `NCT-Full-SameDoc-EN`, `NCT-Human-Random-EN`, `NCT-Human-SameDoc-EN`). Our results show that in general, on 5 out of the 6 English datasets, monolingual `RoBERTa` models rather than monolingual `BERT` models, are the best performing language model on the English datasets. Indicating that size, rather than the NSP pre-training objective, is more effective in capturing narrative coherence for these tasks. Interestingly on 1 out of the 6 datasets multilingual `DeBERTa` yielded the best performing model, showing that multilingual models can compete with and even outperform monolingual models in some scenarios.

Interestingly, we see competitive results comparing the multilingual and monolingual PTLLMs. This might be because the multilingual models such as mDeBERTaV3 are the current state-of-the-art, whereas the monolingual `BERT` and `RoBERTa` models are previous state-of-the-art. However, this also shows the evolution of current state-of-the-art multilingual models compared to previous state-of-the-art monolingual ones and their ability to compete with them. That being said, our results show that the monolingual PTLLMs outperform the multilingual PTLLMs on most datasets. Moreover, we can see that `mDeBERTa` tends to be the best performing multilingual PTLLM, even-though it is a smaller model, compared to `XLM-RoBERTa`, indicating that pre-training objective, rather than model size might be more beneficial. Comparing English to Dutch results, we see a small consistent drop in performance on the Dutch datasets. For the MT-Dutch datasets, this might be due to machine-translation artifacts.

Comparing random to non-random dataset results, we can clearly see the models' struggling to find consistency across the layers, due to increased complexity in the `SameDoc` datasets. A possible explanation for this extra complexity compared to the SCT datasets is: (i) the NCT datasets consists of parts of text excerpts from news articles, therefore the sentences are long(er) ($\approx$26 (NCT)/$\approx$16 (NCT-Dutch) words per sentence compared to $\approx$9 words per sentence for the SCT datasets), and (ii) possibly contain multiple protagonists per story, therefore the contextualized embedding representations possibly need to encode more information about more context. Furthermore, (iii) the correct follow-up sentence is not a story-ending sentence, therefore it can be more flexible in terms of what is (more) correct and what is less/not correct as the follow-up sentence does not necessarily function as a conclusion to the input context. In comparison, the SCT datasets contain short(er) sentences with mostly one protagonist. Whilst also having a final story-ending sentence, providing/needing a more conclusive sequence, relative to the input context sequences.

Comparing English to Dutch results we see that the performance on the Dutch version of the datasets tend to be lower. This might also be due to machine-translation artifacts, so we cannot say with certainty that these models perform worse on Dutch datasets. However, we do see a similar trend comparing the English NCT datasets to the (monolingual, not machine-translated) NCT-Dutch datasets. But, this might due to differences in text-domain, rather than just the difference in language.

Our results indicate that the PTLLMs perform worse when being proxied with the Human variant of a dataset, i.e. only human protagonists, compared to Full. This might indicate that it's harder to encode some sort of linguistic coreference information in the embedding representations about a specific person across temporally and logically connected sequences in a short narrative. Especially when there might be multiple different persons present in such a short narrative. Moreover, the PTLLMs show to struggle more when the difference between the correct and incorrect final sentence is linguistically more subtle. The PTLLMs have no issue performing well on the random datasets, whereas we see a drop, in absolute numbers, compared to the performance on the non-random datasets. For the SCT datasets we see a drop in performance (≈0.2 points), showing that the PTLLMs are able to capture some forms of narrative coherence, with increasing scores as the layers increase. Whereas for the `SameDoc` NCT datasets the drop is larger (≈0.3–0.4 points), with inconsistent, fluctuating scores as the layers increase. The low(er) scores and small(er) differences between the models on the `SameDoc` probing experiments indicate that (i) the PTLLMs fail to capture narrative coherence in these settings and (ii) the results might be noise as we need statistical tests to verify if the models actually follow different trends.

Our probing experiments and datasets better qualify the narrative coherence abilities of PTLLMs. Primarily, our probing results indicate that in a zero-shot setting these PTLLMs perform well under coherent *vs.* random conditions (NCT-Random, NCT-Dutch-Random), with decent performance on short coherent short(er) sentence fictional stories (Story Cloze), whilst struggling with performance and consistency under coherent *vs.* same document conditions (NCT-SameDoc, NCT-Dutch-SameDoc). Fine-tuning these models, on especially the `SameDoc` task, might significantly improve performance. Our results - in absolute terms - range between a minimum of 0.573 (`NCT-Full-Dutch`) to a maximum of 0.915 (`NCT-Human-Random`), based on single layer accuracy score. As described in Chapter 3, our datasets have different levels of complexity for narrative coherence. Not surprisingly, performances on the `SameDoc` NCT datasets, highest complexity, are the lowest because they require a deeper understanding of the context that PTLLMs partially capture. Our probing results on the `SCT-v1.0` dataset suggest that PTLLMs' representations are less sensitive to stylistic bias, with our best results being higher (Δ = 0.059) than those of a comparable approach using SVM and static pre-trained embeddings (Mostafazadeh et al., 2017). The difference with Koto et al. (2021a) is larger (Δ=-0.131), however they use a fine-tuned `ELECTRA` model, using a multilayer perceptron with the `[CLS]` token.

6 CONCLUSION

This work offers a broad investigation of the abilities of six different PTLLMs to encode narrative coherence across sixteen cloze task datasets in English and Dutch, investigating multiple different input settings and different linguistic variations of the final sentence alternative.

In general, our probes indicate that having access to the full text yield better performance when probing for narrative coherence, than extracted single lexical items such as (main) event triggers. Moreover, our probes show that the models fail to capture narrative coherence when using these extracted contextualized representations, indicating that these contextualized representations alone do not contain enough linguistic information to perform well when being probed for narrative coherence. More specifically, our results show that PTLLMs are able to capture narrative coherence in simple cases, when the data consists of short news-article narratives with a random final sentence alternative, without subtle linguistic differences, when having access to the full text. Moreover, PTLLMs are able to capture some narrative coherence when the data consists of short fictional stories with subtle linguistic differences between final sentence alternatives, when having access to the full text. However, the PTLLMs fail to capture narrative coherence when they do not have access to the full text, only (main) event triggers, and when the data consists of short news-article narratives and subtle linguistic differences between final sentence alternatives are present. Monolingual PTLLMs, in general, outperform multilingual PTLLMs. However, the multilingual PTLLMs show competitive results with the monolingual PTLLMs and are able to outperform the monolingual PTLLMs in some scenarios. Furthermore, increased model size, rather than the next sentence prediction (NSP) pre-training objective, seems to yield better performance on this task. Comparing the multilingual PTLLMs' results show that, pre-training objective, rather than size obtain better results when probing for narrative coherence. Lastly, where possible, we have shown that our probe models are competitive with state-of-the-art systems on the same task, indicating that improvements are due either to specific architectures, fine-tuning or extra features capturing additional linguistic information not immediately available in the zero-shot contextual representations.

This work shows that although PTLLMs are able to capture some forms of narrative coherence across temporally and logically connected sequences, narrative coherence remains challenging, especially when subtle linguistic variations, requiring deeper forms of understanding of the context, are present in the final sentence alternatives.

6.1 LIMITATIONS & FUTURE WORK

Future work could focus on expanding this research across more and different languages, such as comparing all Germanic languages and extend this research to Latin or Asian/Arabic languages, to see if similar trends are present. Furthermore, the impact of fine-tuning these models would be interesting to see if it significantly improves performance. Specifically on datasets where a deeper understanding of the context is required to deal with long(er) format sentences and to differentiate between subtle (same document) linguistic variations. Moreover, experimenting with more linguistic variations of possible final sentence alternatives could be explored to get a better understanding of how PTLLMs perform when being challenged by specific linguistic variations (i.e. tense, negation etc.). Also extending this research using more (different) PTLLMs, such as encoders from generative LLMs (such as (m)T5 (Raffel et al., 2020; Xue et al., 2021)), would be insightful to see if similar trends are found across a wider scope of PTLLMs.

Another angle to extend this exploratory research could be to see how PTLLMs react to the distance of a sentence given a context (i.e. given a story of n sentences, use the first n sentences as input, $n+1$ as the correct final sentence, with the next $n+1+n$ sentence as the incorrect final sentence alternative, with increasing distance from the correct $(n+1)$ final sentence). Running the same experiments with increased/-ing input context size would be interesting, to see if these PTLLMs benefit from larger input context or that they struggle with processing increased amounts of information, when choosing between a set of final sentence alternatives given some context (inspired by Liu et al. (2023)).

Lastly, a comparison to static word embeddings on the same tasks would provide more insights into if these contextualized embeddings are better at capturing narrative coherence.

ACKNOWLEDGEMENTS

We thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Hábrók high performance computing cluster.

BIBLIOGRAPHY

Abbott, P. H. (2014). Narrativity. In P. Hühn, J. Pier, W. Schmid, and J. Schönert (Eds.), *the living handbook of narratology*.

Alain, G. and Y. Bengio (2016). Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Angelidis, S., L. Frermann, D. Marcheggiani, R. Blanco, and L. Màrquez (2019, November). Book QA: Stories of challenges and opportunities. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, Hong Kong, China, pp. 78–85. Association for Computational Linguistics.

Belinkov, Y. (2018). *On internal language representations in deep learning: An analysis of machine translation and speech recognition*. Ph. D. thesis, Massachusetts Institute of Technology.

Belinkov, Y. (2022, March). Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics 48*(1), 207–219.

Belinkov, Y. and J. Glass (2019). Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics (TACL) 7*, 49–72.

Bojanowski, P., E. Grave, A. Joulin, and T. Mikolov (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics 5*, 135–146.

Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152.

Caselli, T., I. Dini, and F. Dell'Orletta (2022, October). How about time? probing a multilingual language model for temporal relations. In *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea, pp. 3197–3209. International Committee on Computational Linguistics.

Caselli, T., I. Dini, and F. Dell'Orletta (2022). How about time? probing a multilingual language model for temporal relations. In *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 3197–3209.

Caselli, T. and P. Vossen (2017, August). The event StoryLine corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, Vancouver, Canada, pp. 77–86. Association for Computational Linguistics.

Chambers, N. and D. Jurafsky (2008). Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pp. 789–797.

Chambers, N. and D. Jurafsky (2009). Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pp. 602–610. Association for Computational Linguistics.

Chaturvedi, S., H. Peng, and D. Roth (2017, September). Story comprehension for predicting what happens next. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 1603–1614. Association for Computational Linguistics.

Cho, K., B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Clark, K., M.-T. Luong, Q. V. Le, and C. D. Manning (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov (2019). Unsupervised cross-lingual representation learning at scale. *CoRR abs/1911.02116*.

Conneau, A., G. Kruszewski, G. Lample, L. Barrault, and M. Baroni (2018, July). What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, pp. 2126–2136. Association for Computational Linguistics.

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

de Vries, W., A. van Cranenburgh, and M. Nissim (2020a). What's so special about bert's layers? a closer look at the nlp pipeline in monolingual and multilingual models. *arXiv preprint arXiv:2004.06499*.

de Vries, W., A. van Cranenburgh, and M. Nissim (2020b, November). What's so special about BERT's layers? a closer look at the NLP pipeline in monolingual and multilingual models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online, pp. 4339–4350. Association for Computational Linguistics.

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

Derby, S., P. Miller, and B. Devereux (2021, June). Representation and pre-activation of lexical-semantic knowledge in neural language models. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, Online, pp. 211–221. Association for Computational Linguistics.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2019, June). 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)*, Minneapolis, Minnesota, pp. 4171–4186. Association for Computational Linguistics.

Fisher, W. R. (1984). Narration as a human communication paradigm: The case of public moral argument. *Communications Monographs 51*(1), 1–22.

Fisher, W. R. (1985). The narrative paradigm: An elaboration. *Communications Monographs 52*(4), 347–367.

Goodfellow, I. J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). Generative adversarial networks.

Granroth-Wilding, M. and S. Clark (2016). What happens next? event prediction using a compositional neural network model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 30.

Gupta, A., G. Boleda, M. Baroni, and S. Padó (2015). Distributional vectors encode referential attributes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 12–21.

He, P., J. Gao, and W. Chen (2021). Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

He, P., X. Liu, J. Gao, and W. Chen (2021). Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Hochreiter, S. and J. Schmidhuber (1997). Long short-term memory. *Neural computation 9*(8), 1735–1780.

Howard, J. and S. Ruder (2018, July). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, pp. 328–339. Association for Computational Linguistics.

Huang, Z., W. Xu, and K. Yu (2015). Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Hupkes, D., S. Veldhoen, and W. Zuidema (2018). Visualisation and'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research 61*, 907–926.

Jin, Z., X. Zhang, M. Yu, and L. Huang (2022, December). Probing script knowledge from pre-trained models. In *Proceedings of the Workshop on Unimodal and Multimodal Induction of Linguistic Structures (UM-IoS)*, Abu Dhabi, United Arab Emirates (Hybrid), pp. 87–93. Association for Computational Linguistics.

Junczys-Dowmunt, M., R. Grundkiewicz, T. Dwojak, H. Hoang, K. Heafield, T. Neckermann, F. Seide, U. Germann, A. F. Aji, N. Bogoychev, et al. (2018). Marian: Fast neural machine translation in c++. *arXiv preprint arXiv:1804.00344*.

Köhn, A. (2015). What's in an embedding? analyzing word embeddings through multilingual evaluation.

Koto, F., J. H. Lau, and T. Baldwin (2021a, June). Discourse probing of pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, pp. 3849–3864. Association for Computational Linguistics.

Koto, F., J. H. Lau, and T. Baldwin (2021b). Discourse probing of pretrained language models. *arXiv preprint arXiv:2104.05882*.

Kurfalı, M. and R. Östling (2021, August). Probing multilingual language models for discourse. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, Online, pp. 8–19. Association for Computational Linguistics.

Lal, Y. K., N. Chambers, R. Mooney, and N. Balasubramanian (2021, August). TellMeWhy: A dataset for answering why-questions in narratives. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Online, pp. 596–610. Association for Computational Linguistics.

Liu, F., T. Cohn, and T. Baldwin (2018, July). Narrative modeling with memory chains and semantic supervision. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Melbourne, Australia, pp. 278–284. Association for Computational Linguistics.

Liu, N. F., M. Gardner, Y. Belinkov, M. E. Peters, and N. A. Smith (2019). Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*.

Liu, N. F., K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang (2023). Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

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

Loáiciga, S., A. Beyer, and D. Schlangen (2022, October). New or old? exploring how pre-trained language models represent discourse entities. In *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea, pp. 875–886. International Committee on Computational Linguistics.

Maudslay, R. H., J. Valvoda, T. Pimentel, A. Williams, and R. Cotterell (2020). A tale of a probe and a parser. *arXiv preprint arXiv:2005.01641*.

Miaschi, A., D. Brunato, F. Dell'Orletta, and G. Venturi (2020, December). Linguistic profiling of a neural language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online), pp. 745–756. International Committee on Computational Linguistics.

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Minard, A.-L., M. Speranza, E. Agirre, I. Aldabe, M. van Erp, B. Magnini, G. Rigau, and R. Urizar (2015, June). SemEval-2015 task 4: TimeLine: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, Denver, Colorado, pp. 778–786. Association for Computational Linguistics.

Mirza, P. and S. Tonelli (2016, December). Catena: Causal and temporal relation extraction from natural language texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan, pp. 64–75. The COLING 2016 Organizing Committee.

Mosbach, M., S. Degaetano-Ortlieb, M.-P. Krielke, B. M. Abdullah, and D. Klakow (2020, December). A closer look at linguistic knowledge in masked language models: The case of relative clauses in American English. In *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online), pp. 771–787. International Committee on Computational Linguistics.

Mostafazadeh, N., A. Grealish, N. Chambers, J. Allen, and L. Vanderwende (2016, June). Caters: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events*, San Diego, California, pp. 51–61. Association for Computational Linguistics.

Mostafazadeh, N., M. Roth, A. Louis, N. Chambers, and J. Allen (2017, April). LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, Valencia, Spain, pp. 46–51. Association for Computational Linguistics.

Pandia, L., Y. Cong, and A. Ettinger (2021, November). Pragmatic competence of pre-trained language models through the lens of discourse connectives. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, Online, pp. 367–379. Association for Computational Linguistics.

Pandit, O. and Y. Hou (2021, June). Probing for bridging inference in transformer language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, pp. 4153–4163. Association for Computational Linguistics.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12*, 2825–2830.

Pennington, J., R. Socher, and C. D. Manning (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog 1*(8), 9.

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research 21*(1), 5485–5551.

Rudinger, R., P. Rastogi, F. Ferraro, and B. Van Durme (2015). Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1681–1686.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1985). Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Sadeghi, K. (2014). Phrase cloze: A better measure of reading. *The Reading Matrix 14*(1).

Sancheti, A. and R. Rudinger (2022, July). What do large language models learn about scripts? In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, Seattle, Washington, pp. 1–11. Association for Computational Linguistics.

Sharma, R., J. Allen, O. Bakhshandeh, and N. Mostafazadeh (2018a). Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 752–757.

Sharma, R., J. Allen, O. Bakhshandeh, and N. Mostafazadeh (2018b, July). Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Melbourne, Australia, pp. 752–757. Association for Computational Linguistics.

Sorodoc, I.-T., K. Gulordava, and G. Boleda (2020, July). Probing for referential information in language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, pp. 4177–4189. Association for Computational Linguistics.

Tenney, I., D. Das, and E. Pavlick (2019a). Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Tenney, I., D. Das, and E. Pavlick (2019b, July). BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp. 4593–4601. Association for Computational Linguistics.

Tiedemann, J. (2012). Parallel data, tools and interfaces in opus. In *Lrec*, Volume 2012, pp. 2214–2218. Citeseer.

Tiedemann, J. and S. Thottingal (2020). OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

UzZaman, N., H. Llorens, L. Derczynski, J. Allen, M. Verhagen, and J. Pustejovsky (2013, June). SemEval-2013 task 1: TempEval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, Atlanta, Georgia, USA, pp. 1–9. Association for Computational Linguistics.

Vashishtha, S., A. Poliak, Y. K. Lal, B. Van Durme, and A. S. White (2020, November). Temporal reasoning in natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online, pp. 4070–4078. Association for Computational Linguistics.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017). Attention is all you need. *Advances in neural information processing systems 30*.

Vulić, I., E. M. Ponti, R. Litschko, G. Glavaš, and A. Korhonen (2020, November). Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, pp. 7222–7240. Association for Computational Linguistics.

Weber, N., N. Balasubramanian, and N. Chambers (2018). Event representations with tensor-based compositions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 32.

Wilner, S., D. Woolridge, and M. Glick (2021). Narrative embedding: Recontextualization through attention. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1393–1405.

Xue, L., N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel (2021, June). mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, pp. 483–498. Association for Computational Linguistics.

Yeh, C.-L., B. Loni, M. Hendriks, H. Reinhardt, and A. Schuth (2019). Dpgmedia2019: A dutch news dataset for partisanship detection.

7 APPENDICES

7.1 DETAILED PROBING RESULTS

The following tables illustrate the detailed per layer results of each model in the experiment settings we have described in § 4.2 per model for each dataset, in detail. Best score per layer is in bold, best overall layer score is highlighted in yellow.

7.1.1 English Datasets

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.537	0.53	0.542	0.537	**0.566**	0.554	0.554	0.554	0.563	0.55	0.558	0.561	–
mDeBERTaV3-base	0.54	0.537	0.55	0.554	0.556	0.564	0.567	**0.576**	0.569	0.568	0.573	0.559	–
RoBERTa-base	0.559	0.566	0.572	0.569	0.578	**0.583**	0.578	0.565	0.574	0.578	0.573	0.555	–
BERT-base	0.546	0.552	0.555	0.555	0.56	0.547	0.554	0.564	0.553	0.566	**0.571**	0.564	–
XLM-R-base_Full	0.536	0.537	0.538	0.547	0.564	0.558	0.562	0.557	**0.569**	0.551	0.55	0.561	–
mDeBERTaV3-base_Full	0.535	0.536	0.54	0.564	0.55	0.563	0.561	**0.571**	0.569	0.564	0.565	0.559	–
RoBERTa-base_Full	0.557	0.562	0.569	0.573	0.575	**0.584**	0.576	0.57	0.574	0.573	0.575	0.555	–
BERT-base_Full	0.546	0.555	0.556	0.559	0.559	0.545	0.555	0.559	0.556	0.569	**0.572**	0.569	–
XLM-R-base_EvTr	0.478	0.536	0.539	0.528	0.532	**0.546**	0.542	0.544	0.542	0.539	0.53	0.543	–
mDeBERTaV3-base_EvTr	0.52	0.526	0.53	0.533	0.531	0.525	0.546	0.546	0.552	0.548	0.538	**0.555**	–
RoBERTa-base_EvTr	0.53	0.534	0.532	**0.541**	0.54	0.539	0.536	0.523	0.534	0.532	0.538	0.531	–
BERT-base_EvTr	0.517	0.52	0.516	0.518	0.535	0.542	0.529	0.526	0.539	0.529	0.537	**0.548**	–
XLM-R-base_FullEv	0.491	0.53	0.535	0.528	0.526	**0.554**	0.536	0.545	0.535	0.537	0.532	0.542	–
mDeBERTaV3-base_FullEv	0.517	0.522	0.526	0.527	0.535	0.524	0.544	0.543	**0.549**	0.544	0.543	0.547	–
RoBERTa-base_FullEv	0.533	0.545	0.532	0.535	0.538	**0.546**	0.529	0.527	0.528	0.523	0.541	0.532	–
BERT-base_FullEv	0.513	0.524	0.521	0.521	0.526	0.538	0.527	0.526	0.536	0.527	0.536	**0.547**	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.52
TF-IDF_BASELINE	–												0.535

Table 7.1.1: Per layer per model accuracy (binary classification) for the NCT-Full dataset.

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.788	0.804	0.819	0.84	0.859	0.877	0.875	0.882	**0.883**	0.875	0.881	0.877	–
mDeBERTaV3-base	0.788	0.792	0.808	0.835	0.843	0.853	0.86	0.871	0.881	0.89	**0.893**	0.878	–
RoBERTa-base	0.835	0.859	0.866	0.889	0.896	0.898	0.898	0.896	0.895	**0.905**	0.899	0.888	–
BERT-base	0.798	0.808	0.815	0.821	0.835	0.846	0.859	0.865	**0.869**	0.861	0.867	0.862	–
XLM-R-base_Full	0.802	0.814	0.827	0.847	0.867	0.884	0.881	**0.889**	0.886	0.877	0.88	0.879	–
mDeBERTaV3-base_Full	0.793	0.802	0.816	0.841	0.849	0.858	0.866	0.879	0.884	0.891	**0.901**	0.887	–
RoBERTa-base_Full	0.84	0.863	0.881	0.894	0.901	0.899	0.903	0.894	0.897	**0.904**	0.899	0.89	–
BERT-base_Full	0.805	0.811	0.82	0.825	0.838	0.847	0.867	0.87	0.872	0.871	**0.873**	0.867	–
XLM-R-base_EvTr	0.674	0.687	0.694	0.725	0.746	0.778	0.787	0.788	0.785	**0.789**	0.786	0.785	–
mDeBERTaV3-base_EvTr	0.661	0.683	0.711	0.743	0.759	0.774	0.79	0.817	0.832	**0.838**	0.834	0.8	–
RoBERTa-base_EvTr	0.722	0.726	0.752	0.784	0.816	**0.823**	0.816	0.812	0.809	0.799	0.8	0.799	–
BERT-base_EvTr	0.688	0.7	0.719	0.711	0.742	0.766	0.778	0.78	**0.795**	0.795	0.792	0.786	–
XLM-R-base_FullEv	0.679	0.693	0.702	0.739	0.761	0.788	0.799	**0.8**	0.795	0.79	0.789	0.791	–
mDeBERTaV3-base_FullEv	0.659	0.688	0.711	0.745	0.766	0.78	0.797	0.832	0.835	**0.841**	0.84	0.805	–
RoBERTa-base_FullEv	0.728	0.739	0.767	0.8	0.826	**0.833**	0.83	0.829	0.814	0.814	0.816	0.81	–
BERT-base_FullEv	0.697	0.706	0.715	0.722	0.758	0.772	0.784	0.792	**0.812**	0.806	0.807	0.79	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.79
TF-IDF_BASELINE	–												0.824

Table 7.1.2: Per layer per model accuracy (binary classification) for the NCT-Full Random dataset.

	Layer Score												Score
Model	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.547	0.535	**0.568**	0.543	0.564	0.544	0.561	0.548	0.555	0.568	0.562	0.567	–
mDeBERTaV3-base	0.541	0.546	0.562	0.553	0.568	0.591	0.554	0.582	**0.592**	0.56	0.56	0.567	–
RoBERTa-base	0.568	0.576	0.562	0.562	0.581	0.566	0.567	**0.582**	0.568	0.554	0.572	0.58	–
BERT-base	0.524	0.57	0.549	0.554	**0.572**	0.562	0.547	0.572	0.556	0.53	0.546	0.543	–
XLM-R-base_Full	0.541	0.525	0.556	0.55	0.564	0.553	0.551	0.55	0.549	0.55	0.563	**0.576**	–
mDeBERTaV3-base_Full	0.548	0.56	0.572	0.537	0.559	**0.587**	0.555	0.573	0.583	0.553	0.551	0.568	–
RoBERTa-base_Full	0.568	0.578	0.562	0.583	**0.588**	0.548	0.566	0.585	0.58	0.551	0.555	0.564	–
BERT-base_Full	0.543	0.555	0.547	0.561	**0.586**	0.572	0.562	0.563	0.563	0.533	0.533	0.54	–
XLM-R-base_EvTr	0.497	0.515	0.504	0.52	0.534	0.562	**0.585**	0.578	0.561	0.542	0.54	0.561	–
mDeBERTaV3-base_EvTr	0.49	0.489	0.493	0.521	0.518	0.488	0.505	0.542	0.546	0.537	0.559	**0.56**	–
RoBERTa-base_EvTr	0.536	0.514	0.517	**0.543**	0.531	0.524	0.529	0.518	0.505	0.531	0.531	0.511	–
BERT-base_EvTr	0.483	0.489	0.492	0.505	0.495	0.53	**0.548**	0.535	0.531	0.516	0.496	0.49	–
XLM-R-base_FullEv	0.507	0.533	0.514	0.527	0.54	0.561	**0.572**	0.564	0.542	0.538	0.541	0.567	–
mDeBERTaV3-base_FullEv	0.491	0.49	0.484	0.533	0.521	0.509	0.509	0.534	0.557	0.531	**0.569**	0.56	–
RoBERTa-base_FullEv	0.515	0.522	0.521	0.518	0.536	0.518	**0.537**	0.52	0.51	0.525	0.528	0.507	–
BERT-base_FullEv	0.489	0.482	0.495	0.497	0.505	0.531	0.537	0.538	**0.544**	0.51	0.514	0.48	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.56
TF-IDF_BASELINE	–												0.505

Table 7.1.3: Per layer per model accuracy (binary classification) for the NCT-Human dataset.

	Layer Score												Score
Model	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.782	0.805	0.815	0.827	0.856	0.882	0.891	0.888	**0.893**	0.885	0.888	0.866	–
mDeBERTaV3-base	0.774	0.785	0.811	0.843	0.83	0.857	0.862	0.883	0.892	**0.91**	0.907	0.872	–
RoBERTa-base	0.837	0.858	0.875	0.899	0.901	0.909	0.904	0.899	**0.915**	0.897	0.89	0.859	–
BERT-base	0.769	0.774	0.805	0.808	0.811	0.849	0.857	0.86	0.864	**0.87**	0.864	0.862	–
XLM-R-base_Full	0.782	0.793	0.821	0.824	0.857	0.879	**0.893**	0.888	0.89	0.875	0.882	0.873	–
mDeBERTaV3-base_Full	0.773	0.792	0.817	0.841	0.844	0.857	0.857	0.88	0.882	**0.908**	0.899	0.873	–
RoBERTa-base_Full	0.843	0.869	0.869	0.907	0.908	**0.912**	0.907	0.905	0.911	0.895	0.897	0.873	–
BERT-base_Full	0.768	0.773	0.805	0.806	0.82	0.844	0.859	0.858	**0.864**	0.858	0.857	0.86	–
XLM-R-base_EvTr	0.662	0.692	0.686	0.711	0.723	0.783	0.818	0.808	0.815	**0.82**	0.811	0.795	–
mDeBERTaV3-base_EvTr	0.626	0.645	0.665	0.702	0.697	0.721	0.714	0.791	**0.817**	0.808	0.791	0.74	–
RoBERTa-base_EvTr	0.685	0.711	0.728	0.756	0.786	0.807	**0.812**	0.812	0.8	0.808	0.8	0.78	–
BERT-base_EvTr	0.641	0.651	0.647	0.651	0.684	0.695	0.735	0.747	0.766	0.763	**0.773**	0.74	–
XLM-R-base_FullEv	0.672	0.69	0.695	0.731	0.73	0.769	0.82	0.814	0.812	**0.822**	0.82	0.806	–
mDeBERTaV3-base_FullEv	0.638	0.641	0.663	0.71	0.698	0.723	0.728	0.793	0.804	**0.813**	0.805	0.737	–
RoBERTa-base_FullEv	0.683	0.703	0.716	0.756	0.802	0.802	0.794	**0.811**	0.782	0.806	0.8	0.778	–
BERT-base_FullEv	0.639	0.651	0.653	0.66	0.691	0.708	0.738	0.753	**0.767**	0.766	0.762	0.743	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.82
TF-IDF_BASELINE	–												0.8

Table 7.1.4: Per layer per model accuracy (binary classification) for the NCT-Human Random dataset.

	Layer Score												Score
Model	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.683	0.669	0.681	0.709	0.7	0.695	0.716	**0.722**	0.712	0.705	0.694	0.716	–
mDeBERTaV3-base	0.672	0.666	0.668	0.712	0.7	0.724	0.732	0.726	0.736	**0.738**	0.719	0.704	–
RoBERTa-base	0.701	0.699	0.711	0.719	0.731	0.743	0.745	0.751	0.747	0.753	**0.754**	0.736	–
BERT-base	0.681	0.687	0.676	0.671	0.706	0.707	0.721	**0.733**	0.712	0.707	0.706	0.697	–
XLM-R-base_Full	0.68	0.663	0.683	0.708	0.7	0.69	0.711	**0.722**	0.701	0.707	0.7	0.694	–
mDeBERTaV3-base_Full	0.68	0.683	0.692	0.701	0.695	**0.73**	0.722	0.716	0.73	0.726	0.723	0.706	–
RoBERTa-base_Full	0.7	0.702	0.725	0.714	0.727	0.743	0.734	**0.759**	0.751	0.751	0.756	0.743	–
BERT-base_Full	0.687	0.692	0.687	0.681	0.709	0.708	0.725	**0.731**	0.721	0.715	0.703	0.71	–
XLM-R-base_EvTr	0.653	0.642	0.648	0.66	0.667	0.686	0.689	0.689	0.706	**0.709**	0.708	0.686	–
mDeBERTaV3-base_EvTr	0.636	0.648	0.66	0.656	0.662	0.686	0.689	**0.708**	0.699	0.702	0.694	0.669	–
RoBERTa-base_EvTr	0.669	0.671	0.672	0.692	0.7	0.706	0.711	0.72	0.72	0.716	**0.724**	0.724	–
BERT-base_EvTr	0.667	0.665	0.649	0.665	0.678	0.676	0.688	0.684	0.684	0.698	**0.699**	0.691	–
XLM-R-base_FullEv	0.655	0.648	0.65	0.659	0.661	0.676	0.683	0.696	**0.701**	0.69	0.684	0.683	–
mDeBERTaV3-base_FullEv	0.651	0.647	0.659	0.666	0.662	0.667	0.687	0.701	0.702	**0.708**	0.7	0.662	–
RoBERTa-base_FullEv	0.672	0.675	0.677	0.702	0.69	0.702	0.706	**0.728**	0.72	0.722	0.718	0.716	–
BERT-base_FullEv	0.656	0.661	0.653	0.663	0.672	0.664	0.687	0.686	0.68	**0.697**	0.695	0.684	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.32
TF-IDF_BASELINE	–												0.7

Table 7.1.5: Per layer per model accuracy (binary classification) for the SCT-v1.0 dataset.

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.677	0.671	0.673	0.671	0.692	0.725	0.713	0.706	0.724	**0.728**	0.708	0.712	–
mDeBERTaV3-base	0.65	0.669	0.672	0.687	0.708	0.706	0.701	**0.731**	0.724	0.731	0.706	0.689	–
RoBERTa-base	0.696	0.71	0.704	0.714	0.729	**0.747**	0.739	0.747	0.736	0.745	0.741	0.731	–
BERT-base	0.703	0.686	0.699	0.697	0.692	0.702	**0.728**	0.725	0.721	0.721	0.718	0.71	–
XLM-R-base_Full	0.671	0.673	0.655	0.673	0.682	0.711	0.696	0.711	0.713	**0.731**	0.701	0.706	–
mDeBERTaV3-base_Full	0.662	0.662	0.669	0.701	0.699	0.704	0.696	**0.734**	0.72	0.725	0.694	0.689	–
RoBERTa-base_Full	0.699	0.715	0.71	0.725	0.724	0.742	0.733	0.738	**0.743**	0.738	0.74	0.735	–
BERT-base_Full	0.694	0.68	0.696	0.68	0.697	0.706	0.714	0.726	**0.734**	0.718	0.729	0.696	–
XLM-R-base_EvTr	0.632	0.624	0.641	0.646	0.657	**0.685**	0.67	0.672	0.669	0.683	0.653	0.65	–
mDeBERTaV3-base_EvTr	0.629	0.629	0.646	0.648	0.655	0.65	0.647	**0.692**	0.68	0.677	0.688	0.645	–
RoBERTa-base_EvTr	0.643	0.626	0.665	0.675	0.701	0.708	0.711	0.714	0.704	**0.72**	0.716	0.694	–
BERT-base_EvTr	0.652	0.631	0.625	0.614	0.637	0.663	0.657	0.665	0.667	**0.697**	0.691	0.685	–
XLM-R-base_FullEv	0.636	0.625	0.662	0.642	0.642	0.658	0.65	0.659	0.661	**0.669**	0.663	0.66	–
mDeBERTaV3-base_FullEv	0.627	0.636	0.636	0.643	0.64	0.647	0.645	**0.683**	0.675	0.676	0.682	0.638	–
RoBERTa-base_FullEv	0.652	0.638	0.648	0.669	0.709	0.696	0.704	**0.716**	0.708	0.709	0.713	0.688	–
BERT-base_FullEv	0.636	0.641	0.619	0.611	0.646	0.655	0.648	0.646	0.668	**0.704**	0.696	0.681	–
Random_BASELINE						–							0.5
Lexical_BASELINE						–							0.34
TF-IDF_BASELINE						–							0.691

Table 7.1.6: Per layer per model accuracy 5-fold cross-validation (binary classification) for the SCT-v1.5 dataset.

7.1.2 Dutch Datasets

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.549	0.547	0.55	0.553	**0.562**	0.556	0.551	0.556	0.556	0.554	0.548	0.555	–
mDeBERTaV3-base	0.54	0.539	0.545	0.537	0.552	0.542	0.552	0.558	0.56	**0.573**	0.566	0.542	–
RoBERTV2	0.549	0.546	0.545	0.544	0.551	0.543	0.549	0.553	0.556	**0.566**	0.545	0.552	–
BERTje	0.536	0.551	0.545	0.54	0.542	0.542	0.548	0.554	**0.555**	0.55	0.549	0.549	–
XLM-R-base_Full	0.55	0.544	0.549	0.559	**0.562**	0.558	0.547	0.557	0.55	0.546	0.553	0.552	–
mDeBERTaV3-base_Full	0.54	0.544	0.54	0.539	0.543	0.549	0.552	0.561	0.559	**0.572**	0.56	0.539	–
RoBERTV2_Full	0.541	0.552	0.545	0.539	0.549	0.542	0.546	0.554	0.554	**0.562**	0.544	0.547	–
BERTje_Full	0.54	0.55	0.543	0.544	0.546	0.544	0.548	0.556	**0.557**	0.554	0.546	0.547	–
XLM-R-base_EvTr	0.523	0.53	0.522	0.514	0.525	0.527	**0.544**	0.544	0.541	0.526	0.535	0.532	–
mDeBERTaV3-base_EvTr	0.511	0.515	0.52	0.526	0.532	0.524	0.53	0.532	0.545	**0.547**	0.546	0.539	–
RoBERTV2_EvTr	0.518	0.541	0.531	0.531	0.54	0.522	0.528	0.525	0.531	0.53	**0.538**	0.517	–
BERTje_EvTr	0.517	0.518	0.522	0.515	0.514	0.518	0.528	0.536	**0.537**	0.522	0.526	0.53	–
XLM-R-base_FullEv	0.49	0.532	0.521	0.515	0.53	0.536	0.544	0.54	**0.539**	0.532	0.528	0.523	–
mDeBERTaV3-base_FullEv	0.511	0.509	0.519	0.53	0.529	0.528	0.533	0.534	**0.546**	0.539	0.544	0.538	–
RoBERTV2_FullEv	0.523	0.538	**0.539**	0.529	0.535	0.52	0.526	0.531	0.534	0.531	0.539	0.512	–
BERTje_FullEv	0.513	0.518	0.522	0.516	0.514	0.519	0.532	**0.542**	0.532	0.52	0.529	0.529	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.49
TF-IDF_BASELINE	–												0.542

Table 7.1.7: Per layer per model accuracy (binary classification) for the `NCT-Full` Dutch dataset.

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.76	0.768	0.79	0.805	0.825	0.831	0.841	**0.849**	0.846	0.844	0.846	0.845	–
mDeBERTaV3-base	0.745	0.752	0.773	0.784	0.801	0.813	0.826	0.851	0.854	**0.863**	0.862	0.837	–
RoBERTV2	0.782	0.797	0.788	0.802	0.799	0.814	0.813	0.817	0.818	**0.822**	0.821	0.821	–
BERTje	0.775	0.798	0.802	0.819	0.829	0.829	0.83	0.851	0.851	0.857	**0.862**	0.858	–
XLM-R-base_Full	0.759	0.776	0.797	0.812	0.835	0.839	0.852	0.852	**0.859**	0.853	0.851	0.85	–
mDeBERTaV3-base_Full	0.752	0.761	0.784	0.788	0.81	0.815	0.835	0.859	0.861	**0.873**	0.866	0.845	–
RoBERTV2_Full	0.787	0.798	0.796	0.806	0.808	0.82	0.824	**0.832**	0.827	0.826	0.827	0.827	–
BERTje_Full	0.784	0.803	0.814	0.825	0.831	0.836	0.84	0.856	0.852	0.864	0.862	**0.868**	–
XLM-R-base_EvTr	0.666	0.684	0.691	0.718	0.744	0.781	0.788	0.78	**0.797**	0.789	0.792	0.783	–
mDeBERTaV3-base_EvTr	0.635	0.643	0.645	0.683	0.698	0.727	0.758	0.776	0.79	**0.797**	0.787	0.754	–
RoBERTV2_EvTr	0.703	0.693	0.7	0.71	0.705	0.72	0.72	0.719	0.716	0.713	0.728	**0.729**	–
BERTje_EvTr	0.696	0.718	0.728	0.72	0.736	0.744	0.752	0.753	0.765	0.787	**0.797**	0.792	–
XLM-R-base_FullEv	0.676	0.685	0.695	0.718	0.749	0.783	0.804	0.795	**0.799**	0.798	0.797	0.782	–
mDeBERTaV3-base_FullEv	0.628	0.639	0.649	0.689	0.712	0.735	0.757	0.787	0.794	**0.808**	0.792	0.758	–
RoBERTV2_FullEv	0.704	0.706	0.712	0.712	0.72	0.731	0.735	0.726	0.735	0.732	**0.747**	0.74	–
BERTje_FullEv	0.692	0.727	0.73	0.733	0.738	0.755	0.763	0.769	0.776	0.797	**0.807**	0.802	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.73
TF-IDF_BASELINE	–												0.802

Table 7.1.8: Per layer per model accuracy (binary classification) for the `NCT-Full` Random Dutch dataset.

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.549	0.536	0.548	**0.563**	0.553	0.511	0.537	0.538	0.537	0.546	0.535	0.544	–
mDeBERTaV3-base	0.547	0.535	**0.566**	0.555	0.538	0.536	0.509	0.543	0.555	0.542	0.555	0.548	–
RoBERTV2	0.543	0.551	0.56	0.55	0.534	0.516	0.568	0.548	0.583	0.556	0.549	**0.575**	–
BERTje	0.53	0.529	0.54	0.525	0.563	0.568	**0.572**	0.564	0.55	0.55	0.549	0.534	–
XLM-R-base_Full	0.54	0.531	0.53	0.54	**0.557**	0.518	0.524	0.547	0.542	0.554	0.542	0.55	–
mDeBERTaV3-base_Full	0.549	0.538	0.56	0.547	0.528	0.549	0.529	0.548	0.562	0.55	**0.566**	0.542	–
RoBERTV2_Full	0.543	0.543	0.564	0.561	0.533	0.527	0.57	0.536	**0.575**	0.546	0.547	0.574	–
BERTje_Full	0.536	0.534	0.542	0.536	0.563	0.569	**0.576**	0.54	0.535	0.559	0.549	0.537	–
XLM-R-base_EvTr	0.525	0.515	0.527	0.533	0.534	0.549	0.525	0.516	**0.555**	0.52	0.525	0.537	–
mDeBERTaV3-base_EvTr	0.511	0.491	0.498	0.469	0.521	0.535	0.509	**0.559**	0.541	0.521	0.531	0.52	–
RoBERTV2_EvTr	0.498	0.542	0.529	0.541	**0.562**	0.531	0.518	0.544	0.518	0.544	0.541	0.528	–
BERTje_EvTr	0.54	0.514	0.546	0.51	0.475	0.497	0.509	0.544	0.535	0.54	**0.559**	0.536	–
XLM-R-base_FullEv	0.529	0.496	0.521	0.534	0.514	**0.559**	0.529	0.531	0.54	0.527	0.528	0.527	–
mDeBERTaV3-base_FullEv	0.48	0.486	0.504	0.476	0.502	0.528	0.544	**0.56**	0.54	0.515	0.55	0.508	–
RoBERTV2_FullEv	0.533	0.523	0.524	0.525	**0.556**	0.529	0.515	0.546	0.512	0.527	0.529	0.52	–
BERTje_FullEv	0.529	0.53	0.538	0.522	0.452	0.473	0.48	0.544	0.52	0.54	**0.559**	0.546	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.52
TF-IDF_BASELINE	–												0.537

Table 7.1.9: Per layer per model accuracy (binary classification) for the `NCT-Human` Dutch dataset.

Model	Layer Score												Score
	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.711	0.733	0.759	0.76	0.821	0.844	0.845	**0.849**	0.847	0.828	0.844	0.831	–
mDeBERTaV3-base	0.721	0.731	0.764	0.776	0.783	0.826	0.851	0.864	0.867	0.879	**0.883**	0.808	–
RoBERTV2	0.772	0.775	0.774	0.769	0.778	0.811	0.819	0.822	0.812	0.824	**0.831**	0.811	–
BERTje	0.766	0.787	0.798	0.811	0.843	0.83	0.821	0.834	0.824	0.859	**0.865**	0.84	–
XLM-R-base_Full	0.733	0.749	0.763	0.767	0.827	0.852	0.84	**0.854**	0.843	0.828	0.844	0.821	–
mDeBERTaV3-base_Full	0.731	0.731	0.767	0.775	0.792	0.828	0.844	0.849	0.867	**0.88**	0.871	0.807	–
RoBERTV2_Full	0.756	0.78	0.773	0.775	0.786	0.813	0.813	0.827	0.806	0.827	**0.83**	0.824	–
BERTje_Full	0.756	0.796	0.809	0.819	0.843	0.824	0.818	0.841	0.84	**0.857**	0.846	0.846	–
XLM-R-base_EvTr	0.622	0.646	0.65	0.67	0.714	0.772	0.762	0.78	**0.793**	0.783	0.772	0.762	–
mDeBERTaV3-base_EvTr	0.588	0.599	0.614	0.604	0.653	0.675	0.725	0.749	0.722	**0.785**	0.757	0.673	–
RoBERTV2_EvTr	0.645	0.647	0.658	0.675	0.666	0.682	0.672	0.667	0.695	0.692	**0.721**	0.708	–
BERTje_EvTr	0.647	0.693	0.683	0.679	0.68	0.671	0.692	0.69	0.718	0.767	**0.781**	0.76	–
XLM-R-base_FullEv	0.625	0.651	0.662	0.669	0.707	0.767	0.768	0.78	**0.782**	0.8	0.778	0.772	–
mDeBERTaV3-base_FullEv	0.602	0.613	0.619	0.624	0.658	0.685	0.728	0.76	0.741	**0.78**	0.742	0.665	–
RoBERTV2_FullEv	0.641	0.653	0.654	0.69	0.666	0.68	0.695	0.684	0.703	0.708	**0.724**	0.722	–
BERTje_FullEv	0.644	0.688	0.684	0.679	0.685	0.675	0.696	0.68	0.72	0.764	**0.786**	0.774	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.76
TF-IDF_BASELINE	–												0.789

Table 7.1.10: Per layer per model accuracy (binary classification) for the `NCT-Human` Random Dutch dataset.

Model	Layer Score												Score
	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.487	0.536	0.519	0.544	0.58	**0.595**	0.57	0.547	0.58	0.57	0.574	0.565	–
mDeBERTaV3-base	0.502	0.501	0.543	0.519	0.56	0.589	0.556	0.557	**0.604**	0.559	0.55	0.503	–
RoBERTV2	0.546	0.545	0.527	0.543	0.559	**0.568**	0.55	0.542	0.531	0.532	0.547	0.559	–
BERTje	0.542	0.563	0.538	0.549	0.536	0.54	0.554	0.575	0.563	0.577	**0.599**	0.582	–
XLM-R-base_Full	0.456	0.525	0.508	0.542	0.582	**0.587**	0.567	0.551	0.575	0.551	0.559	0.549	–
mDeBERTaV3-base_Full	0.503	0.474	0.54	0.518	0.565	**0.581**	0.54	0.557	0.599	0.556	0.567	0.522	–
RoBERTV2_Full	**0.561**	0.544	0.527	0.506	0.552	0.544	0.547	0.54	0.523	0.529	0.523	0.553	–
BERTje_Full	0.546	0.549	0.547	0.53	0.536	0.509	0.532	0.57	0.564	0.556	0.588	**0.589**	–
XLM-R-base_EvTr	0.529	0.542	0.503	0.49	0.502	0.481	0.531	**0.543**	0.494	0.531	0.523	0.531	–
mDeBERTaV3-base_EvTr	0.456	0.449	0.465	0.505	0.532	0.503	0.517	**0.547**	0.525	0.547	0.529	0.505	–
RoBERTV2_EvTr	0.464	0.455	0.47	0.489	0.502	0.482	0.522	0.494	0.487	0.465	0.504	**0.523**	–
BERTje_EvTr	0.526	0.516	0.531	0.517	0.533	0.52	0.52	0.49	0.524	0.54	**0.55**	0.545	–
XLM-R-base_FullEv	0.549	0.517	0.505	0.477	0.496	0.482	0.516	**0.553**	0.48	0.53	0.537	0.543	–
mDeBERTaV3-base_FullEv	0.433	0.462	0.462	0.524	0.532	0.523	0.518	0.537	0.526	**0.556**	0.513	0.509	–
RoBERTV2_FullEv	0.478	0.456	0.492	0.51	0.496	0.505	0.519	0.488	0.475	0.502	0.524	**0.531**	–
BERTje_FullEv	0.503	0.484	0.544	0.51	0.522	0.508	0.529	0.509	0.532	0.529	**0.544**	0.538	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.43
TF-IDF_BASELINE	–												0.496

Table 7.1.11: Per layer per model accuracy 5-fold cross-validation (binary classification) for the `NCT-Full-Dutch` dataset.

Model	Layer Score												Score
	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.673	0.698	0.717	0.76	0.772	0.806	0.814	0.832	0.833	0.835	**0.837**	0.82	–
mDeBERTaV3-base	0.662	0.69	0.695	0.733	0.768	0.779	0.804	0.816	0.827	0.826	0.843	**0.85**	–
RoBERTV2	0.754	0.784	0.782	0.786	0.795	0.791	0.796	0.789	0.8	0.805	**0.821**	0.8	–
BERTje	0.745	0.768	0.77	0.779	0.804	0.812	0.839	0.863	0.865	**0.875**	0.863	0.867	–
XLM-R-base_Full	0.671	0.691	0.719	0.763	0.778	0.799	0.819	0.819	**0.836**	0.822	0.832	0.822	–
mDeBERTaV3-base_Full	0.655	0.684	0.689	0.735	0.768	0.779	0.813	0.812	0.826	0.836	**0.841**	0.84	–
RoBERTV2_Full	0.761	0.789	0.789	0.788	0.792	0.789	0.792	0.804	0.796	0.804	**0.805**	0.805	–
BERTje_Full	0.754	0.761	0.76	0.793	0.816	0.808	0.837	0.862	0.871	**0.881**	0.878	0.858	–
XLM-R-base_EvTr	0.609	0.604	0.623	0.675	0.709	0.735	0.744	0.73	0.76	0.76	**0.761**	0.747	–
mDeBERTaV3-base_EvTr	0.593	0.596	0.628	0.655	0.681	0.689	0.683	0.749	0.756	0.761	**0.767**	0.738	–
RoBERTV2_EvTr	0.63	0.661	0.683	0.687	0.674	0.683	0.695	0.677	0.673	0.687	**0.713**	0.684	–
BERTje_EvTr	0.639	0.694	0.694	0.691	0.712	0.712	0.737	0.724	0.765	0.779	**0.812**	0.786	–
XLM-R-base_FullEv	0.596	0.593	0.63	0.67	0.701	0.719	0.739	0.738	0.752	**0.768**	0.749	0.752	–
mDeBERTaV3-base_FullEv	0.579	0.592	0.62	0.641	0.687	0.697	0.685	0.732	0.753	**0.78**	0.77	0.751	–
RoBERTV2_FullEv	0.636	0.669	0.688	0.699	0.676	0.681	0.69	0.674	0.665	0.684	**0.709**	0.704	–
BERTje_FullEv	0.649	0.697	0.681	0.688	0.71	0.716	0.742	0.733	0.761	0.78	**0.816**	0.795	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.62
TF-IDF_BASELINE	–												0.751

Table 7.1.12: Per layer per model accuracy 5-fold cross-validation (binary classification) for the `NCT-Full-Dutch` Random dataset.

Model	Layer Score												Score
	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.499	0.466	0.482	0.527	0.524	0.533	0.469	**0.553**	0.54	0.501	0.514	0.505	–
mDeBERTaV3-base	0.467	0.486	0.549	0.502	0.514	0.466	0.517	0.463	0.483	**0.54**	0.422	0.508	–
RoBERTV2	0.441	0.501	0.517	0.486	**0.537**	0.534	0.512	0.502	0.425	0.428	0.495	0.476	–
BERTje	0.591	**0.607**	0.585	0.572	0.547	0.544	0.514	0.563	0.531	0.47	0.55	0.543	–
XLM-R-base_Full	0.489	0.485	0.492	0.527	0.527	0.524	0.479	**0.572**	0.511	0.508	0.508	0.527	–
mDeBERTaV3-base_Full	0.495	0.518	0.527	**0.534**	0.527	0.495	0.518	0.479	0.467	0.521	0.428	0.492	–
RoBERTV2_Full	0.463	0.498	0.495	0.476	**0.54**	0.527	0.508	0.463	0.418	0.412	0.476	0.47	–
BERTje_Full	**0.601**	0.601	0.579	0.562	0.534	0.537	0.498	0.553	0.534	0.489	0.527	0.553	–
XLM-R-base_EvTr	0.482	0.45	0.425	0.476	0.476	0.499	0.461	0.514	0.428	0.47	0.54	**0.565**	–
mDeBERTaV3-base_EvTr	0.473	0.498	0.521	0.501	0.511	0.537	0.457	0.479	0.527	**0.54**	0.502	0.527	–
RoBERTV2_EvTr	0.444	0.511	0.496	0.511	0.467	0.512	0.47	0.502	0.53	**0.537**	0.438	0.476	–
BERTje_EvTr	0.498	0.518	0.55	0.537	0.514	0.515	0.54	0.524	0.511	0.549	0.489	**0.581**	–
XLM-R-base_FullEv	0.454	0.489	0.419	0.46	0.495	0.476	0.448	0.501	0.457	0.489	0.527	**0.53**	–
mDeBERTaV3-base_FullEv	0.473	0.482	0.518	0.466	0.511	0.54	0.479	0.486	0.508	**0.559**	0.499	0.546	–
RoBERTV2_FullEv	0.463	0.518	0.464	0.498	0.473	0.502	0.482	0.479	**0.55**	0.547	0.425	0.469	–
BERTje_FullEv	0.489	0.486	0.543	0.512	0.524	0.505	0.543	0.514	0.492	0.546	0.469	**0.556**	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.41
TF-IDF_BASELINE	–												0.415

Table 7.1.13: Per layer per model accuracy 5-fold cross-validation (binary classification) for the NCT-Human-Dutch dataset.

Model	Layer Score												Score
	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.719	0.731	0.748	0.78	0.805	0.844	0.844	0.84	0.847	**0.859**	0.841	0.808	–
mDeBERTaV3-base	0.674	0.681	0.671	0.725	0.757	0.783	0.78	0.792	**0.856**	0.853	0.837	0.843	–
RoBERTV2	0.744	0.757	0.77	0.767	0.805	**0.821**	0.812	0.773	0.738	0.78	0.786	0.729	–
BERTje	0.741	0.751	0.764	0.773	0.821	0.827	0.843	0.856	0.834	**0.863**	0.847	0.818	–
XLM-R-base_Full	0.719	0.735	0.747	0.786	0.805	0.831	**0.856**	0.831	0.85	0.853	0.844	0.786	–
mDeBERTaV3-base_Full	0.665	0.687	0.668	0.716	0.77	0.78	0.783	0.789	0.853	**0.856**	0.843	0.85	–
RoBERTV2_Full	0.741	0.77	0.764	0.763	0.786	**0.799**	0.799	0.78	0.738	0.776	0.773	0.741	–
BERTje_Full	0.744	0.732	0.748	0.786	0.808	0.818	0.846	**0.863**	0.837	0.847	0.85	0.831	–
XLM-R-base_EvTr	0.642	0.633	0.62	0.655	0.639	0.69	0.709	0.741	**0.779**	0.741	0.725	0.703	–
mDeBERTaV3-base_EvTr	0.514	0.537	0.556	0.55	0.553	0.556	0.581	0.636	0.7	**0.716**	0.709	0.671	–
RoBERTV2_EvTr	0.62	0.597	0.639	0.639	0.632	0.604	0.617	0.661	0.568	0.623	**0.668**	0.633	–
BERTje_EvTr	0.53	0.642	0.645	0.652	0.687	0.687	0.677	0.69	0.738	0.757	**0.767**	0.738	–
XLM-R-base_FullEv	0.626	0.626	0.626	0.661	0.648	0.684	0.709	0.751	**0.77**	0.757	0.725	0.725	–
mDeBERTaV3-base_FullEv	0.527	0.556	0.588	0.553	0.553	0.572	0.562	0.626	0.706	**0.738**	0.709	0.677	–
RoBERTV2_FullEv	0.629	0.601	0.629	0.629	0.613	0.591	0.629	0.668	0.581	0.613	**0.681**	0.636	–
BERTje_FullEv	0.546	0.652	0.658	0.674	0.709	0.719	0.687	0.709	0.709	0.757	**0.773**	0.741	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.62
TF-IDF_BASELINE	–												0.665

Table 7.1.14: Per layer per model accuracy 5-fold cross-validation (binary classification) for the NCT-Human-Dutch Random dataset.

Model	Layer Score												Score
	1	2	3	4	5	6	7	8	9	10	11	12	
XLM-R-base	0.66	0.664	0.651	0.66	0.68	0.679	**0.696**	0.691	0.694	0.681	0.685	0.683	–
mDeBERTaV3-base	0.665	0.687	0.688	0.685	0.702	0.691	0.707	**0.72**	0.706	0.719	0.695	0.702	–
RoBERTV2	0.683	0.699	0.696	0.704	0.689	0.693	0.692	0.702	**0.721**	0.707	0.714	0.712	–
BERTje	0.678	0.685	0.673	0.675	0.671	0.683	0.684	0.686	0.69	0.696	**0.7**	0.695	–
XLM-R-base_Full	0.668	0.661	0.672	0.665	0.689	0.692	**0.703**	0.695	0.696	0.694	0.691	0.697	–
mDeBERTaV3-base_Full	0.665	0.67	0.684	0.681	0.693	0.696	0.7	**0.723**	0.707	0.713	0.707	0.707	–
RoBERTV2_Full	0.679	0.698	0.7	0.701	0.692	0.678	0.701	0.69	**0.719**	0.712	0.706	0.717	–
BERTje_Full	0.679	0.687	0.682	0.681	0.678	0.686	0.68	0.689	0.693	0.692	0.689	**0.694**	–
XLM-R-base_EvTr	0.629	0.624	0.614	0.625	0.64	0.649	0.66	0.672	0.672	0.671	**0.68**	0.676	–
mDeBERTaV3-base_EvTr	0.611	0.62	0.616	0.614	0.665	0.65	0.657	0.664	0.678	0.692	**0.703**	0.649	–
RoBERTV2_EvTr	0.648	0.645	0.66	0.657	0.662	0.665	0.662	0.658	0.663	0.663	**0.684**	0.669	–
BERTje_EvTr	0.639	0.649	0.658	0.66	0.666	0.653	0.658	0.663	0.658	0.655	0.683	**0.684**	–
XLM-R-base_FullEv	0.631	0.625	0.622	0.629	0.638	0.647	0.655	0.68	0.664	0.664	**0.681**	0.68	–
mDeBERTaV3-base_FullEv	0.605	0.63	0.614	0.604	0.658	0.651	0.663	0.679	0.667	**0.693**	0.688	0.635	–
RoBERTV2_FullEv	0.641	0.642	0.647	0.646	0.656	0.657	0.651	0.66	0.661	0.67	**0.677**	0.666	–
BERTje_FullEv	0.644	0.647	0.662	0.652	0.643	0.661	0.649	0.659	0.651	0.658	**0.687**	0.678	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.29
TF-IDF_BASELINE	–												0.688

Table 7.1.15: Per layer per model accuracy (binary classification) for the SCT-v1.0 Dutch dataset.

	Layer Score												Score
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
XLM-R-base	0.653	0.666	0.661	**0.68**	0.668	0.684	0.68	0.649	0.669	0.67	0.673	0.657	–
mDeBERTaV3-base	0.654	0.655	0.646	0.674	0.667	0.67	0.688	0.698	**0.703**	0.673	0.671	0.669	–
RoBERTV2	0.671	0.667	0.677	0.659	0.685	0.66	0.655	0.654	0.666	0.668	0.682	**0.694**	–
BERTje	0.668	0.654	0.659	0.669	0.664	0.646	0.66	0.678	0.666	0.667	0.671	**0.685**	–
XLM-R-base_Full	0.655	0.659	0.661	0.669	0.669	0.669	**0.682**	0.648	0.666	0.673	0.671	0.662	–
mDeBERTaV3-base_Full	0.649	0.655	0.658	0.651	0.664	0.662	0.673	0.685	**0.697**	0.671	0.664	0.664	–
RoBERTV2_Full	0.663	0.658	0.667	0.659	0.678	0.655	0.651	0.66	0.666	0.674	**0.694**	0.693	–
BERTje_Full	0.666	0.654	0.661	0.668	0.671	0.653	0.668	**0.678**	0.672	0.662	0.671	0.669	–
XLM-R-base_EvTr	0.617	0.627	0.599	0.615	0.626	0.617	0.631	0.639	0.634	0.657	**0.661**	0.629	–
mDeBERTaV3-base_EvTr	0.583	0.611	0.609	0.617	0.631	0.628	0.647	0.664	**0.668**	0.659	0.673	0.631	–
RoBERTV2_EvTr	0.645	0.629	0.627	0.642	0.622	0.629	0.624	0.64	0.65	0.643	0.672	**0.678**	–
BERTje_EvTr	0.632	**0.646**	0.644	0.62	0.617	0.623	0.629	0.646	0.622	0.638	0.632	0.631	–
XLM-R-base_FullEv	0.61	0.618	0.588	0.605	0.634	0.615	0.622	0.639	0.641	0.64	**0.654**	0.622	–
mDeBERTaV3-base_FullEv	0.583	0.603	0.605	0.612	0.623	0.622	0.659	0.659	0.654	0.657	**0.661**	0.631	–
RoBERTV2_FullEv	0.636	0.642	0.634	0.638	0.61	0.623	0.613	0.643	0.646	0.641	0.662	**0.675**	–
BERTje_FullEv	0.624	**0.65**	0.643	0.634	0.611	0.63	0.632	0.633	0.637	0.635	0.646	0.629	–
Random_BASELINE	–												0.5
Lexical_BASELINE	–												0.31
TF-IDF_BASELINE	–												0.672

Table 7.1.16: Per layer per model accuracy 5-fold cross-validation (binary classification) for the SCT-v1.5 Dutch dataset.