What's so special about BERT's layers? A closer look at the NLP pipeline in monolingual and multilingual models

Wietse de Vries, **Andreas van Cranenburgh**, and **Malvina Nissim**

CLCG, University of Groningen, The Netherlands

{wietse.de.vries,a.w.van.cranenburgh,m.nissim}@rug.nl

Abstract

Peeking into the inner workings of BERT has shown that its layers resemble the classical NLP pipeline, with progressively more complex tasks being concentrated in later layers. To investigate to what extent these results also hold for a language other than English, we probe a Dutch BERT-based model and the multilingual BERT model for Dutch NLP tasks. In addition, through a deeper analysis of part-of-speech tagging, we show that also within a given task, information is spread over different parts of the network and the pipeline might not be as neat as it seems. Each layer has different specialisations, so that it may be more useful to combine information from different layers, instead of selecting a single one based on the best overall performance.

1 Introduction and Background

Natural Language Processing is now dominated by transformer-based models (Vaswani et al., 2017), like BERT (Devlin et al., 2019), a model trained on predicting masked tokens and relations between sentences. BERT's impact is so strong that we already talk about 'BERTology' (Rogers et al., 2020).

In addition to using BERT in NLP tasks and end applications, research has also been done *on* BERT, especially to reveal what linguistic information is available in different parts of the model. This is done, e.g., investigating what BERT's attention heads might be attending to (Clark et al., 2019), or looking at its internal vector representations using so-called probing (or diagnostic) classifiers (Tenney et al., 2019a). It has been noted that BERT progressively acquires linguistic information roughly in the same the order of the classic language processing pipeline (Tenney et al., 2019b,a): surface features are expressed in lower layers, syntactic features more in middle layers and semantic ones in higher layers (Jawahar et al., 2019). So, for example, information on part-of-speech appears to be acquired earlier than on coreference.

Most work dedicated to understanding the inner workings of BERT has focused on English, though non-English BERT models do exist, in two forms. One is a multilingual model (Devlin et al., 2019, mBERT), which is trained on Wikipedia dumps of 104 different languages. The other one is a series of monolingual BERTs (Polignano et al., 2019; Le et al., 2019; Virtanen et al., 2019; Martin et al., 2019; de Vries et al., 2019, among others). As expected, also the non-English monolingual BERT models achieve state-of-the-art results on a variety of NLP tasks, and mostly outperform the multilingual model on common NLP tasks (Nozza et al., 2020). Nevertheless, mBERT performs surprisingly well on zero-shot POS tagging and Named Entity Recognition (NER), as well as on cross-lingual model transfer (Pires et al., 2019).

If these results imply that the inner workings of other monolingual BERTs and of mBERT are the same as BERT's is not yet known. Also not known is how *homogeneous* layer specialisation is: through general performance of, e.g., POS tagging, we see a peak at a given layer, but we do not know how specialisation actually evolves across the whole model. This work investigates such issues.

Contributions Using probing classifiers for four tasks on six datasets for a monolingual Dutch model and for mBERT, we observe that (i) these models roughly exhibit the same classic pipeline observed for the original BERT, suggesting this is a general feature of BERT-based models; (ii) the most informative mBERT layers are consistently earlier layers than in monolingual models, indicating an inherent task-independent difference between the two models. Through a deeper analysis of POS tagging, we also show that (iii) the picture of a neatly ordered NLP pipeline is not

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4339–4350
November 16 - 20, 2020. ©2020 Association for Computational Linguistics

completely correct, since information appears to be more spread across layers than suggested by the performance peak at a given layer.

The full source code is publicly available on Github[1].

2 Approach

We run two kinds of analyses.

The first is aimed at a rather high level comparison of the performance of a monolingual (Dutch) BERT model (BERTje, de Vries et al. 2019) and multilingual BERT (mBERT) on a variety of tasks at different levels of linguistic complexity (POS tagging, dependency parsing, named entity recognition, and coreference resolution; see Section 2.2), with attention to what happens at different layers.

The second is an in-depth analysis of the performance of BERTje and mBERT on part-of-speech tagging. The reason behind this is that looking at global performance over a given task does not provide enough information on what is actually learned by different layers of the model *within* that task. POS tagging lends itself well for this type of layerwise evaluation. First, because it is a low level task for which relatively little real-world knowledge is required. Second, because analysis of single tags is straightforward since it is done at a token level. Third, because POS tagging contains both easy and difficult cases that depend on surrounding context. Some words are more ambiguous than others, and some classes are open whereas others are closed. Token ambiguity may for instance be an important factor for differences between a monolingual and a multilingual model since the latter has to deal with more homographs, due to the co-presence of multiple languages.

Section 2.3 describes how these analyses can be performed in practice using the probes.

2.1 Experimental setup

Our method for measuring task performance at different layers is based on the edge probing approach of Tenney et al. (2019a,b). Edge probing is a method to evaluate how well linguistic information can be extracted from a pre-trained encoder. Separate trained classifiers on the outputs of Transformer layers in BERT can reveal which layers contain most information for a particular task.

The inputs of the probing classifiers are embeddings extracted from the lexical layer (layer 0) and each Transformer layer (layers 1 up to 12) from either the pre-trained BERTje or mBERT model. Embeddings of token spans are extracted from these full sentence or document embeddings and those spans are used as probe model inputs. The probing classifiers are trained to predict task labels based on span representations using an LSTM layer for tokens that require multiple WordPieces.[2]

For each model, layer and task we train two probes: a single layer based probe and a scalar mixing probe. The single layer probe uses a single pre-trained Transformer layer output as its input, whereas the scalar mixing probes use a weighted sum of the target layer and preceding layers.

2.2 Tasks and Data

We train the probing classifiers on six datasets with four different tasks, chosen to represent linguistic layers of abstraction.[3] For POS tagging and dependency parsing, the LassySmall and Alpino datasets from Universal Dependencies (UD) v2.5 (Zeman et al., 2019) are used with provided splits. For Named Entity Recognition, we use the Dutch portion of the CoNLL-2002 NER dataset (Tjong Kim Sang, 2002) with the provided splits. Finally, we use the coreference annotations of the SoNaR-1 corpus (Delaere et al., 2009) for coreference, with document level training (80%), validation (10%) and testing (10%) splits.

2.3 Analysis

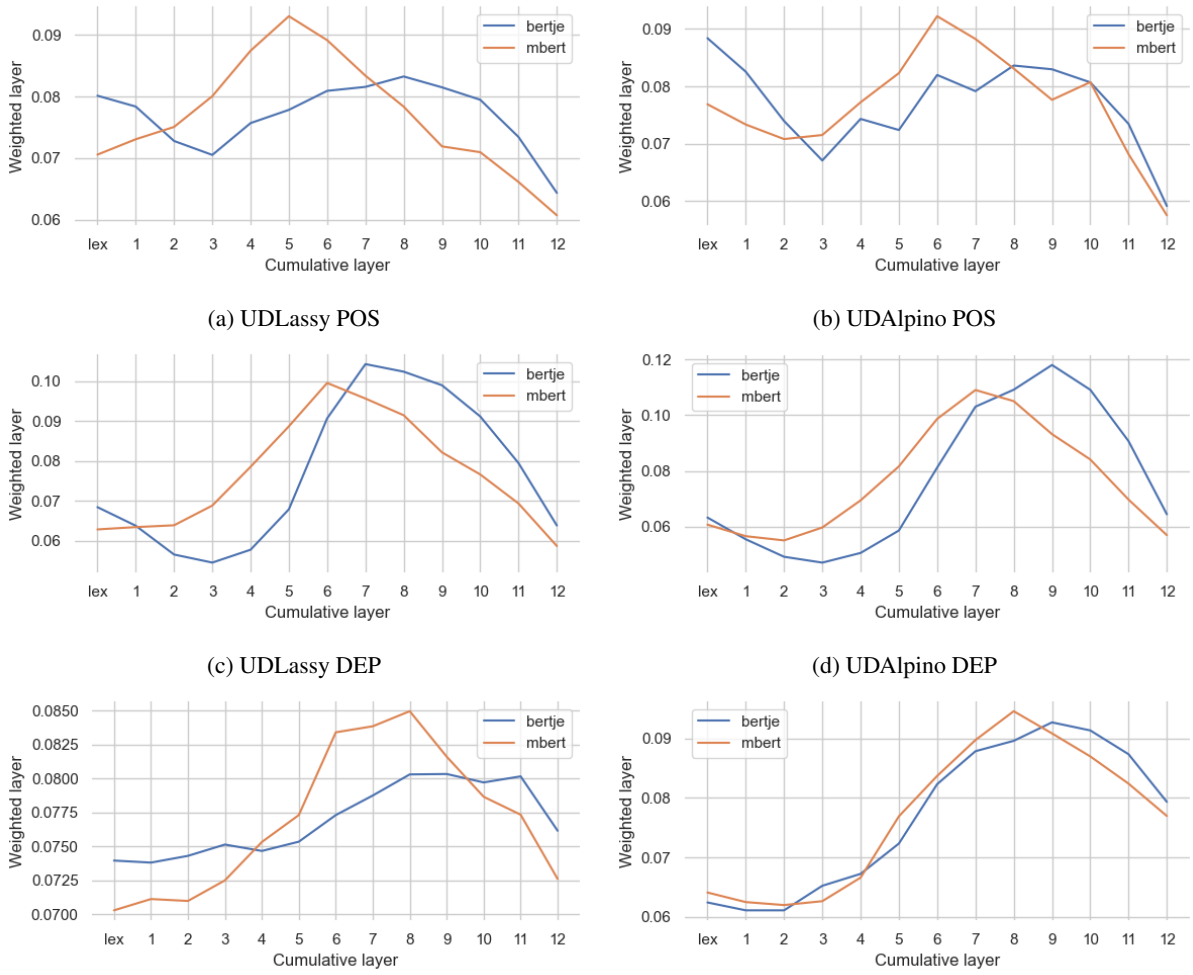
We perform a series of analyses aimed at creating a picture of what happens inside of BERTje and mBERT. Initial overall analyses of the tasks are done with the scalar mixing probes as well as the single layer probes for each of the six tasks.

First, weights that the scalar mixing probes give to each pre-trained model layer are compared (Section 3.1). Layers that get larger scalar mixing weights may be considered to be more informative than lower weight layers for a particular task (Tenney et al., 2019a). It does not have to be the case that the most informative layers are at the same position in the model since an interaction between layers in different positions may be even more informative. Therefore, we compare layer weights between tasks and pre-trained models. The two

[1] https://github.com/wietsedv/bertje/tree/master/probing

[2] See Tenney et al. (2019a) for technical details on the classifier architecture. Our hyper-parameters are in Appendix B.

[3] Details on size, splits, and processing are in Appendix A.



(a) UDLassy POS

(b) UDAlpino POS

(c) UDLassy DEP

(d) UDAlpino DEP

(e) CoNLL-2002 NER

(f) SoNaR Coref

Figure 1: Scalar mixing weights for each pre-trained model and each task. Highlights: The sorted weights form clean curves; BERTje makes more use of lexical embeddings; Weights decrease at final layers; mBERT peaks earlier than BERTje; POS and DEP results are consistent across datasets.

different data sources for POS tagging and dependency parsing will give an indication about stability of these weight distributions across datasets and within tasks. These weights are solely based on training data, so they may not represent the exact layer importance for unseen data.

Second, we compare overall prediction scores of the probes on unseen test data for each task (Section 3.2). Through this, we can observe at what stage models peak for what task, and where monolingual and multilingual models might differ. The accuracy deltas between layers for scalar mixing probes will give an indication about which layers add information that was not present in all previous layers combined. For these probes, deltas should be positive if information is added and zero if a layer is uninformative.

Third, we take a closer look at POS tagging (Section 4). The previous analyses reveal information about the amount of task-relevant information that is present in each layer, but POS tagging can require different kinds of abstraction for different labels, so that POS performance might be non-homogeneous across layers. Specifically, we (i) compare layerwise performance for each tag and the groups of open and closed class POS tags; (ii) investigate whether information is lost, learned or relearned within the model by combining probe predictions for each individual token; and (iii) check the most frequent confusions between tags to better understand the causes of errors.

3 Analysis over all tasks

First, the weights of the scalar mixing models are compared in order to see which layer combinations are most informative. These weights are tuned

solely on the training data so they give no indication about layer importance for unseen data. Second, we compare overall prediction scores of the probes on unseen test data for each of the tasks.

3.1 Layer weights

Figure 1 shows the scalar mixing weights of the full scalar mixing probes. We highlight a few important patterns that are consistent between tasks, and suggest possible explanations for what we observe, in particular regarding the differences between BERTje and mBERT.

The sorted weights form clean curves. The probing classifier is ignorant about ordering of layers when the weights are tuned. Nevertheless the sorted weights mostly show clean curves. The clean curves indicate that embedding of useful information for these tasks is gradually added and removed by the transformer models. This also confirms that our probing model is actually sensitive to these gradual changes in the embeddings.

BERTje makes more use of lexical embeddings. The curves in Figure 1 show that the probes for BERTje give higher weights to the first layer than the mBERT probes. This suggests that the pre-trained context-independent lexical embeddings of BERTje are more informative for these tasks than those of mBERT. This makes sense because mBERT word pieces are shared between languages, so there is more word piece level lexical ambiguity in mBERT than BERTje.

The exception to this pattern is the SoNaR coreference task, where the difference between mBERT and BERTje is small. Establishing whether two spans of text corefer requires more context-dependent information in addition to lexical embeddings, whereas the other tasks contain examples where context is not always required. BERTje does not rely on the lexical layer more strongly than on subsequent layers for this task.

Weights decrease at final layers. If the transformer layers continually add information, the final layer would contain most information. However, information actually decreases after peaking in layers 5 to 9. The reason may be that the actual output of the model should be roughly the same as the original input. Therefore generalisations are discarded in favour of representations that map back to actual word pieces. Generalisations may lead to information loss if they do not correspond to our target tasks, because original information may become less accessible after generalisation. The first and last lexical layers contain most token identity information. If the probes did not benefit from learned language model representations, we would observe that these layers are the most important to solve the tasks. However, the weight peaks that we see in between the lexical layers suggest that the language models contain generalisations that are informative for the given tasks.

mBERT peaks earlier than BERTje. The weight peak for the mBERT probes is always in an earlier layer than the peaks of equivalent BERTje probes. These peaks do not correspond with center measures in BERT probing scalar mixing weights of Tenney et al. (2019a), since single center measures only correspond with peaks if the distribution is roughly normal.

This might suggest differing priorities during pre-training. Generally, BERTje's weights start to decrease somewhere in the second half of the layers whereas mBERT's peaks are closer to the center. This suggests that BERTje uses more layers to generalise than to instantiate back to tokens. The large vocabulary and variety of languages in mBERT may require mBERT to start instantiating earlier with an equal amount of generalisation and instantiation as a result.

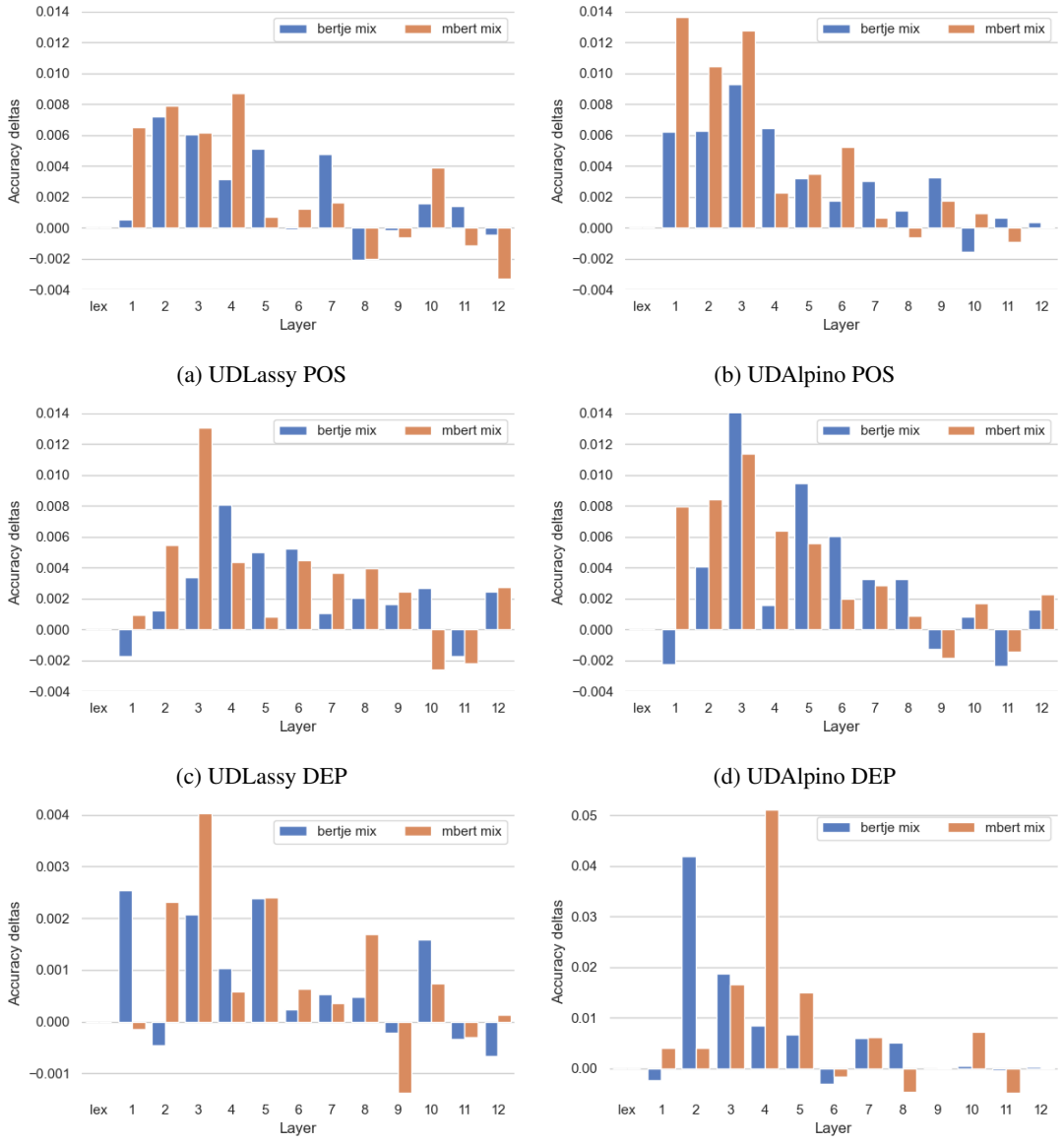
POS and DEP results are consistent across datasets. The UDLassy and UDAlpino datasets contain equivalent annotations, but the data originates from different text genres. Their POS curves in Figure 1a and 1b and their DEP curves in Figure 1c and 1d are however mostly the same. This indicates that the probes are sensitive to the task and the input embeddings, but not overly sensitive to the specific data that the probes are trained on.

3.2 Prediction scores

Figure 2 shows deltas of accuracy scores compared to the preceding layer based on test predictions. The minimum absolute accuracy scores for each task range from 0.630 (SoNaR Coref) to 0.979 (CoNLL-2002 NER) and the maximum accuracy scores per task range from 0.729 (SoNaR Coref) to 0.991 (CoNLL-2002 NER).[4]

Intuitively, positive deltas in the mixing results in Figure 2 indicate that the introduced layer contains new information that was not present in any

[4] Accuracy deltas for single layer probes are in Appendix C.



(a) UDLassy POS

(b) UDAlpino POS

(c) UDLassy DEP

(d) UDAlpino DEP

(e) CoNLL-2002 NER

(f) SoNaR Coref

Figure 2: Accuracy deltas for cumulative introduction of layers with scalar mixing probes. Positive values indicate that these layers contain new task-specific information. Some negative values in later layers suggest overfitting.

preceding layers, whereas zero-deltas indicate that the new layer is completely uninformative. Ideally, the accuracy deltas would never be negative since the probe of layer N has access to information from all layers up to N. Negative deltas with cumulative introduction of layers to the probes suggest that the probes sometimes overfit to training data. Otherwise, these deltas should always be zero or higher. Scalar mixing weights of layers that correspond with these uninformative negative delta layers should be lower in order to reduce their effect on the predictions. Figure 1 shows that negative accuracy deltas mainly correspond with negative weight slopes. Therefore, the effects in Figure 1 may be stronger in optimally performing probes.

The general pattern in the scalar mixing accuracy deltas in Figure 2 is that deltas are positive in earlier layers and improvement stops for the last layers. This fits with the decreasing weights for the last layers in the full scalar mixing model (Figure 1).

One important difference between the layer mixing probes and the single layer probes is that single layer probes sometimes show negative accuracy deltas while the corresponding accuracy delta is positive for the mixing probe. Positive mixing probe deltas suggest that new information is introduced or made more accessible, whereas the negative single layer deltas suggest that some infor-

mation is lost or has been made less accessible by the language model. Intuitively, this indicates that some information is sacrificed in order to make place for new information in the embedding. If that is the case, the actual probe prediction mistakes may change between layers even if overall accuracy scores stay the same.

Analysis of scalar mixing weights or accuracy on the whole test data only gives an indication of the sum of information for a task. However, a more fine-grained error analysis is required to give any indication about what information is retrievable in which layer and what information becomes harder to identify.

4 In-depth analysis for POS tagging

Layer-wise task performance and scalar mixing weights give information about overall information density for a task.

For POS tagging, maximum performance and largest scalar mixing weights are assigned to layers 5 to 9 for the pre-trained models, but this does not tell the whole story. Indeed, probes can make different types of errors for different layers and models, because the models may clarify or lose information between layers. Moreover, different examples and labels within a task may rely on information from different layers.

We want to give a more thorough view of what BERTje and mBERT learn and whether information becomes unidentifiable between layers as well as whether BERTje and mBERT make the same mistakes. Therefore, we evaluate the errors of the UDLassy POS predictions with single layer probes.

We do this analysis on POS predictions because this task stays closest to the lexical level of embedding that the models are pre-trained for, but also rely on context and generalisation for optimal performance. We focus on UDLassy data rather than UDAlpino because the differences between the accuracy deltas of scalar mixing models and single layer models appears higher for UDLassy. This would suggest a larger shift in mistakes.

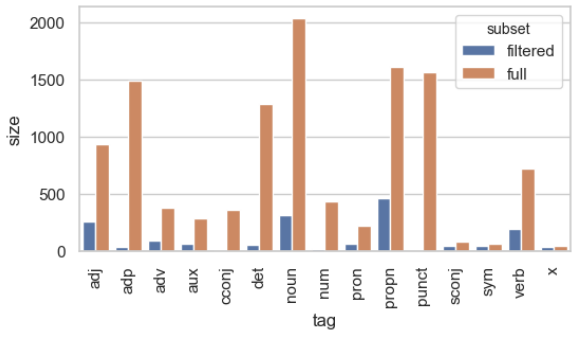


Figure 3: Distributions of POS tags in the full test set as well as the filtered test set. The filtered distribution is not equivalent to the original distribution because some common tags are relatively easy.

The following analysis is done on the predictions of the 13 single layer BERTje probes and the 13 single layer mBERT probes. POS tagging is not difficult for all tokens, so for 85% of the test data all 26 probes predict the correct tag. In order to focus on errors, we perform all analyses using the subset of the tokens that have an incorrect prediction by at least one of the probes. This amounts to 1,720 tokens. The original test data distribution as well as the filtered distribution are shown in Figure 3.

Note that the filtered data distribution does not correspond to the original distribution since some tags are easier to recognise than others. For instance, proper nouns are over-represented in our analysis set whereas adpositions and punctuation are underrepresented. This is not a problem since we are explicitly interested in the mistakes and difficult cases and not in overall performance.

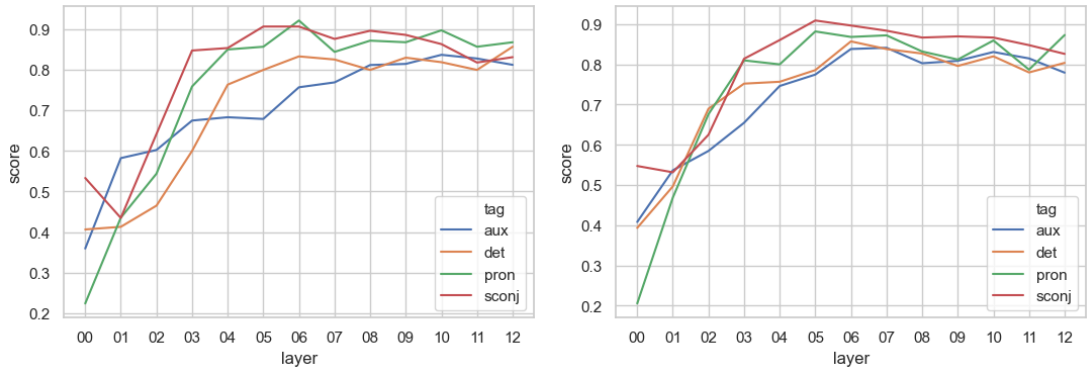
4.1 Accuracies per POS tag

Figures 4 and 5 show the F1 scores per POS tag per layer for the single layer probe predictions.

POS tags are grouped in aggregates based on whether they are considered to be closed categories (Figure 4) or open categories (Figure 5) according to the Universal Dependencies guidelines. There are six POS tags with relatively low average performance, which also have random fluctuations in per layer performance. Therefore, *adp*, *cconj*, *punct*, *num*, *sym* and *x* are left out of Figures 4 and 5.

Figure 4 shows that closed class POS tags seem to be learned by the pre-trained models and not lost in later layers. On average, their scores increase for the first six layers, indicating that the probe uses learned information to identify these tags. After reaching top performance, the probe performance does not really decrease, rather it plateaus. Only the subordinating conjunction class seems to show some decline. There is remarkably little difference between BERTje and mBERT for these classes.

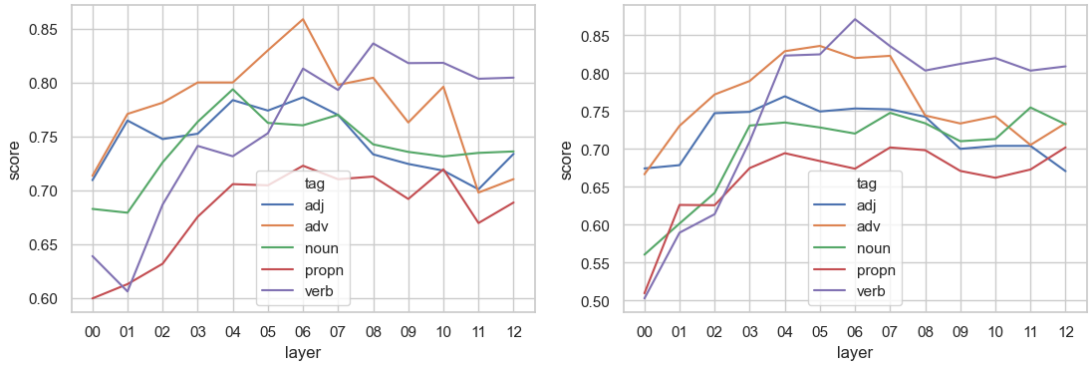
Figure 5 shows the tag F1 scores for open class POS tags. Contrary to the closed classes, the mean scores on open classes do seem to decline in later layers. Within the closed classes there are three



(a) BERTje closed class POS tags

(b) mBERT closed class POS tags

Figure 4: F1 scores per closed class POS tag per layer for BERTje and mBERT. Closed class performance stabilises around the sixth layers and does not significantly decrease.



(a) BERTje open class POS tags

(b) mBERT open class POS tags

Figure 5: F1 scores per open class POS tag per layer for BERTje and mBERT. Except for verbs, performances decrease in later layers. This indicates that these tag representations become hard to distinguish in later layers.

different patterns. Nouns and proper nouns are learned quickly and stay relatively stable. This is especially true for mBERT. For BERTje, the scores for (proper) nouns seem to decline somewhat after reaching a peak. Verbs keep improving for more layers than (proper) nouns. Apparently, recognition of verbs is something that is resolved later in the pre-trained models. Finally, adjectives and adverbs show an actual decline in performance, since these two tags become hard to distinguish from each other, or possibly other tags, in later layers.

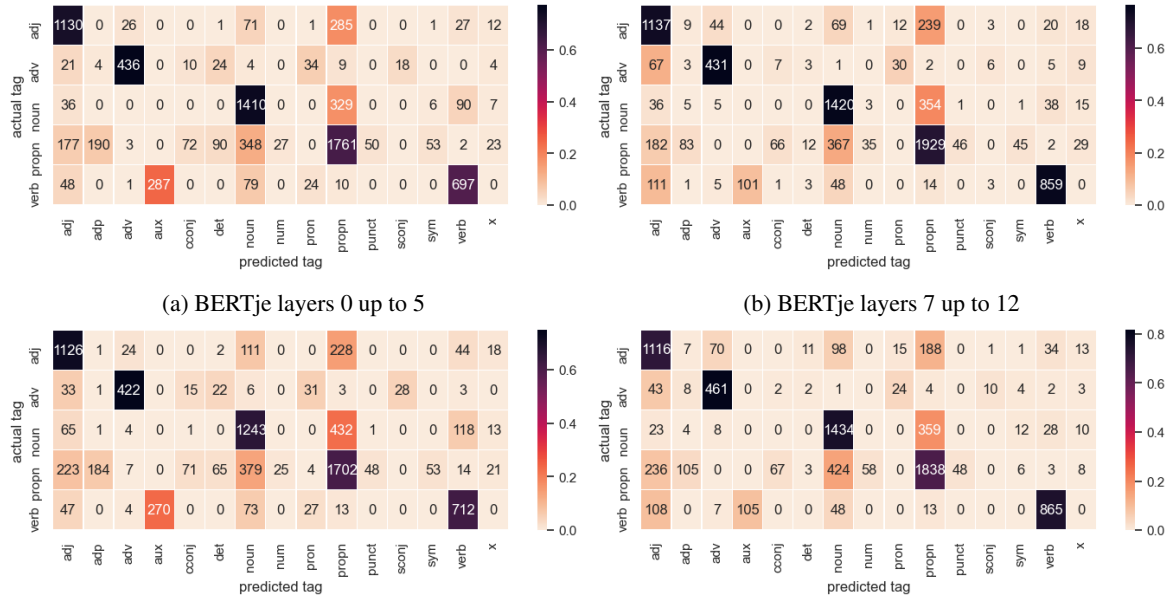
4.2 Confusion between tags

The previous figures give an indication about which POS tags are learned by pre-trained models based on context and which tags become unidentifiable, but they do not give an indication about changes in tag confusion. Figure 5 shows that overall single layer performance of open class words peaks in layer 6 for BERTje and layer 6 is also included in the peak layers for mBERT.

To illustrate whether biases and confusions change after this peak, we compare the summed confusion matrices from the six layers before and the six layers after layer 6. These confusion matrices (Figure 6) show that there are many similarities between BERTje and mBERT with respect to the confusions that are learned or lost.

Decrease in error counts between the first half and the second half of the models suggests that differentiation between tags is learned, whereas increase in errors suggests information loss. For instance verbs and adverbs are more often misclassified as determiners in the first than in the second half. Similarly, proper nouns are confused a lot more often with auxiliary verbs or pronouns in the first half than in the second half.

Those differences suggest that discrimination between these tags is learned by both models. However, nouns and proper nouns are confused with adjectives a lot more often in the second than in the first half.



(a) BERTje layers 0 up to 5

(b) BERTje layers 7 up to 12

(c) mBERT layers 0 up to 5

(d) mBERT layers 7 up to 12

Figure 6: Total confusions of open class POS tags before and after the middle. Confusions are very similar between BERTje and mBERT, but some confusions change between first and last layers.

4.3 Example errors

BERTje and mBERT do not always make the same mistakes, nor are the same mistakes made in each layer. For many tokens, the probes make incorrect predictions for the first layer(s), but start making correct predictions in later layers, which indicates that learned information is used. Often, these error patterns are similar between BERTje and mBERT. The following are examples of differences:

(1) Max Rood — minister van Binnenlandse Zaken , kabinet - Van **Agt** III
[Max Rood — minister of Internal Affairs , cabinet - Van **Agt** III]

(2) **Federale** Regering
[**Federal** Government]

(3) Het ontplooiingsliberalisme stelde de vrije **maar** verantwoordelijke mens centraal.
[The self-development liberalism put the free **but** responsible man central.]

(4) **Reeds** in het begin van de 20ste eeuw . . .
[**Already** in the beginning of the 20th century]

(5) . . . het **Duitstalig** taalgebied . . .
[. . . the **German** language-area . . .]

(6) . . . de Keltische **stammen** in het gebied . . .
[. . . the Celtic **tribes** in the area . . .]

In (1), mBERT initially tags the proper noun “Agt” as verb. In (2) BERTje initially tags the adjective “Federale” as proper noun. Both classifications are incorrect guesses, but with additional context both pre-trained models correctly identify this proper noun in later layers. A different pattern of errors is that the probes make correct predictions based on the first or last layer, but some mistakes for layers in between. In (3) the conjunction “maar” (but) receives the tag adv in several layers instead of the correct tag “cconj”. BERTje makes this mistake in layer 4, 5, and 10; mBERT makes it in layers 3 to 7. It happens relatively often that all BERTje probes assign correct labels, but mBERT goes from incorrect to correct. These mistakes are typically resolved in the first layer of mBERT, suggesting such errors are easily resolvable with a little bit of context; see (4) for an example.

There are also a lot of examples where mBERT probes are always correct, but BERTje probes make a mistake somewhere in the middle. It may be the case that these examples are resolvable with and without context but that the internal representations of BERTje get generalised based on non-POS properties. In (5) the adjective “Duitstalig” gets confused with proper noun in layers 4, 5, 7, 8 and 9, but in the layers before and after BERTje probes get it correct. Semantically it is reasonable to think that “Duitstalig” has proper noun-like properties. Finally, (6) is an example where BERTje is always

correct but mBERT makes a mistake in the middle somewhere. The word "stammen" should be a noun but mBERT sometimes thinks it is a verb.

5 Conclusion

Our results show that BERTje and mBERT exhibit a pipeline-like behaviour along tasks similar to what has previously been shown for English.

Tenney et al. (2019a) observed that the pipeline order is roughly first POS tagging, then named entity recognition, then dependency parsing and coreference resolution. Our results suggest that BERTje encodes these pipeline tasks in a similar order. Scalar mixing weights show that there is not a single layer that contains all important information because the weight curves show peaks and valleys. This suggests that useful task information is distributed between layers. Generally, the most informative layers are located early in the second half of the pre-trained models. As an additional note, because we ran the model on different datasets for the same task, we can assess stability across datasets. We observe that POS tagging and dependency parsing results are consistent, suggesting that the probes are sensitive to the task and the embeddings, but not overly sensitive to the specific data that they are trained on.

The main task differences between the monolingual BERTje model and the multilingual mBERT model are that BERTje probes make more use of the lexical embedding layer than the mBERT probes and the most important layers of BERTje are mostly later layers than those of mBERT.

Semantically rich POS tags like nouns and adjectives become harder to identify in later layers (Figure 5) and confusions mainly happen between semantically rich open categories (Figure 6). This suggests that semantic content is more important than POS discriminating features for final token predictions. So even if the POS abstraction is not readily present in the lexical layer nor in the final token prediction layer, POS tag information is still found in middle layer generalisations. POS tagging is a part of what the pre-trained models learn, but different tag abstractions are present in different layers. Therefore, feature-based use of these models should not use the output of a *single best* layer. It would be better to combine the outputs of multiple or all layers in order to retrieve all learned information that is relevant for a downstream task. However, actual fine-tuning of pre-trained language models should still be a preferred approach.

In sum, our results show that 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. We observed this for POS tagging, but it is still unclear how information within tasks is distributed in these models for other tasks. Moreover, it would be interesting to investigate alternative probing strategies in order to better disentangle what pertains to the model itself from what is specific to a given probing strategy. Lastly, it is an open question how well linguistic properties are embedded within large pre-trained language models for non Indo-European languages.

References

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. In *Proceedings of BlackboxNLP*, pages 276–286.

Isabelle Delaere, Veronique Hoste, and Paola Monachesi. 2009. Cultivating trees: Adding several semantic layers to the Lassy treebank in SoNaR. In *Proceedings of TLT*, pages 135–146.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of ACL*, pages 3651–3657.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2019. FlauBERT: Unsupervised language model pre-training for French. arXiv:1912.05372.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2019. CamemBERT: a Tasty French Language Model. arXiv:1911.03894.

Gertjan van Noord, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. 2013. Large scale syntactic annotation of written Dutch: Lassy. In Peter Spyns and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch: Results by the STEVIN programme*, pages 147–164. Springer Berlin Heidelberg, Berlin, Heidelberg.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2020. What the [MASK]? Making sense of language-specific BERT models. arXiv:2003.02912.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of ACL*, pages 4996–5001.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In *Proceedings of CLiC-it*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. arXiv:2002.12327.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of ACL*, pages 4593–4601.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. In *Proceedings of ICLR*.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*, pages 5998–6008.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: BERT for Finnish. arXiv:1912.07076.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT model. arXiv:1912.09582.

Daniel Zeman, Joakim Nivre, et al. 2019. Universal dependencies 2.5. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

A Data

This is a more detailed description of the data and data preparation that the probing classifiers are trained and tested on.

For token level classification tasks like POS tagging, the input span is the range of word pieces that form a single token. For other tasks that use multi-word expressions, like named entity recognition, the spans can be longer than single tokens. Dependency parsing and coreference resolution are not flat token classification tasks but edge prediction tasks. Therefore the probing model can also predict edge labels if two spans are given. The task specific input and output representations are described below. Table 1 shows the sizes of our training datasets and Table 2 shows the data sizes of our test data. Validation data sizes are nearly the same as test data.

Part-of-speech (POS) tagging For POS tagging, two datasets from Universal Dependencies (UD) v2.5 (Zeman et al., 2019) are used. These two datasets are the LassySmall (UDv2.5 LassySmall POS) and the Alpino (UD Alpino POS) datasets, both of which consist of documents from the Lassy Small corpus (van Noord et al., 2013). The UD-LassySmall data consists of Wikipedia articles whereas the UD-Alpino data originates from news articles. Universal POS tags are used with 16 coarse lexical categories[5]. Both datasets have predefined train, validation and test splits.

Dependency (DEP) parsing For dependency parsing, the same same sources with the same splits are used as for POS tagging: UD-LassySmall and UD-Alpino from UD-v2.5. For uniformity across tasks, the probing classifiers are not trained for attachment but for edge labeling. For each edge in a sentence, the head token is used as one span and the full child sequence is used as the other span. The child span is not a single token since a child forms a semantic unit together with its sub-children. For instance, a child span can be "A small child" with a head token "plays" where "plays" is the actual head of "child" in the dependency tree. The semantics of a dependency relationship may be distributed among the tokens within the child tree. The probing classifier is trained to predict which of the 37 UD syntactic relations is the correct one between the head and child span. Predefined splits are used for training, validating, and testing.

Named Entity Recognition (NER) For Named Entity Recognition, we use the Dutch portion of the CoNLL-2002 NER dataset (Tjong Kim Sang, 2002), which contains BIO-encoded named entity annotations for newspaper articles with four

[5] https://universaldependencies.org/u/pos/

task	# sents	# tokens	# examples	# labels
UDLassy POS	5,787	75,165	75,165	16
UDLassy DEP	5,787	75,165	69,293	34
UDAlpino POS	12,264	185,999	185,999	16
UDAlpino DEP	12,264	185,999	173,619	34
CoNLL-2002 NER	15,806	202,644	114,288	5
SoNaR Coref NER	46,969	773,968	139,005	2

Table 1: Description of our training data.

task	# sents	# tokens	# examples	# labels
UDLassy POS	875	11,581	11,581	16
UDLassy DEP	875	11,581	10,681	34
UDAlpino POS	596	11,053	11,053	16
UDAlpino DEP	596	11,053	10,450	34
CoNLL-2002 NER	5,195	68,875	38,488	5
SoNaR Coref NER	5,094	96,705	17,720	2

Table 2: Description of our test data. All validation data is in the same order of magnitude as test data.

classes: persons, organisations, locations and miscellaneous. Spans for full entities are used as inputs for the probing classifier with the entity class as target label. The non-entity tokens are used as negative samples (*O* label) with random span lengths of one to three tokens. The existing train, validation (test1) and test (test2) splits are used.

Coreference (Coref) resolution For coreference resolution, the coreference annotations from the SoNaR-1 corpus (Delaere et al., 2009) are used. There are no pre-defined splits for training and testing, so a random set of 10% of the documents is used for validation and 10% for testing. The splitting is done at document level, so all sentences from the same document are present in the same split. The coreference task is framed as a binary classification task where two spans of tokens are either coreferential or they are not. Because referents are often mentioned in multiple sentences, embeddings are extracted from the pre-trained models with concatenated sentences, until the token limit of 512 tokens is reached. Half of the examples are coreferential strings and half are random referents that do not corefer. Positive examples are sampled from all possible coreferring spans, whereas negative samples can be any non-coreferring expressions. The data contains annotations for spans of potentially referring expressions including singletons, so spans in negative examples are not limited to expressions that are coreferential with another span.

B Probe hyper-parameters

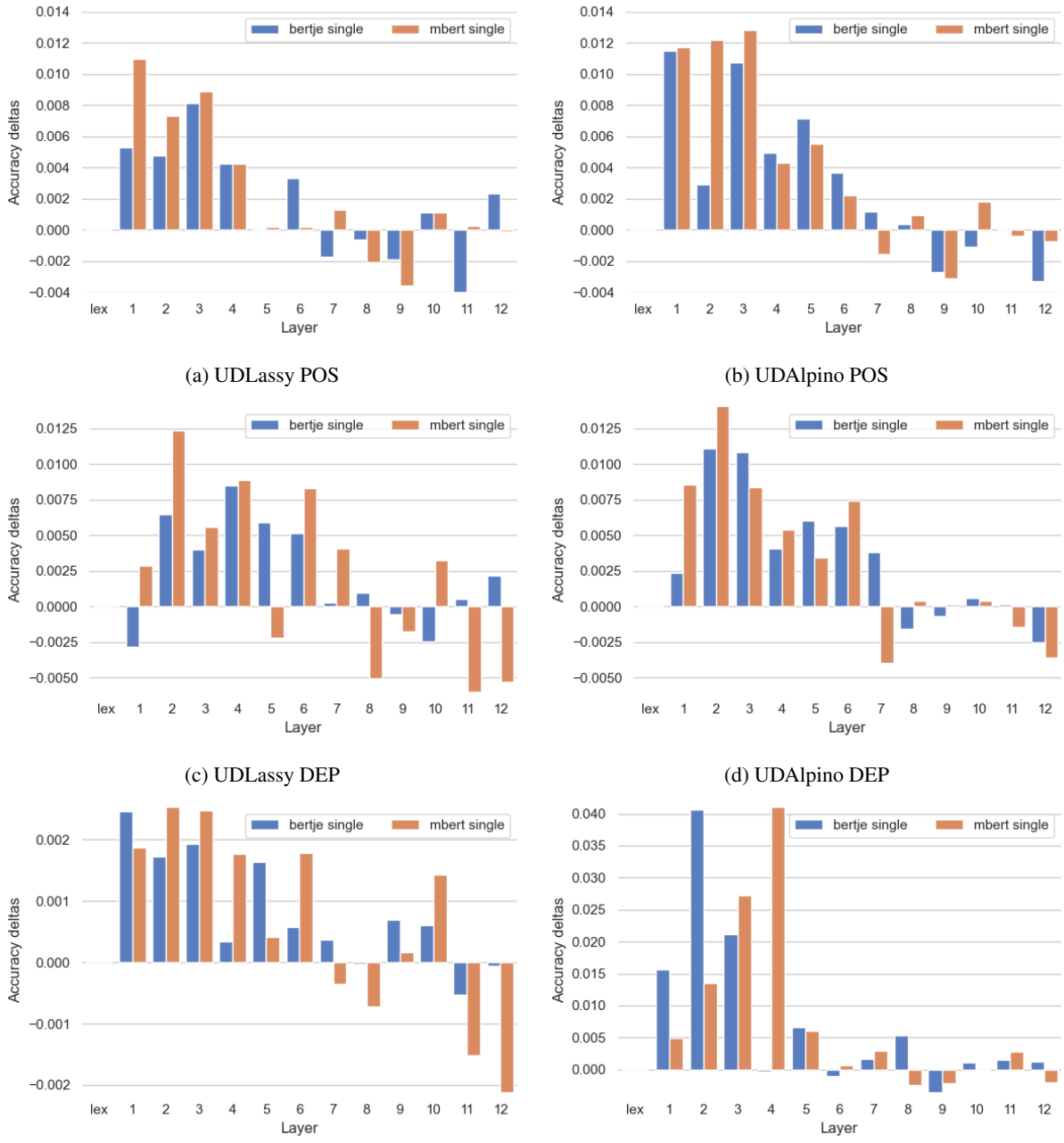
The probing classifiers use the following hyper-parameters:

- Input size: 768 (embedding size of the pre-trained models)
- Hidden layer size: 256
- Number of bidirectional LSTM layers: 2 (for span representations)
- Dropout:
  - Input layer: 0.2
  - Recurrent layers: 0.3
  - Other layers: 0.2

This model is trained with the Adam optimisation algorithm with a learning rate of 0.0001 and weight decay of 0.01. Training is done in mini-batches of 32 examples with evaluation on validation data after every 1000 batches. Training stops when validation loss has not decreased for 20 steps.

C Probe accuracies

The paper includes accuracy deltas for scalar mixing probes for each task. Figure 7 shows the equivalent accuracy deltas for single layer probes.



(a) UDLassy POS

(b) UDAlpino POS

(c) UDLassy DEP

(d) UDAlpino DEP

(e) CoNLL-2002 NER

(f) SoNaR Coref

Figure 7: Accuracy deltas for single layer probes. The general pattern is that the deltas are positive in the earlier layers and improvement stops for the last layers.