Assessing Word Importance Using Models Trained for Semantic Tasks

Dávid Javorský[1] and **Ondřej Bojar**[1] and **François Yvon**[2]

[1]Charles University, Faculty of Mathematics and Physics, Prague, Czechia
[2]Sorbonne Université, CNRS, ISIR, Paris, France
{javorsky,bojar}@ufal.mff.cuni.cz francois.yvon@cnrs.fr

Abstract

Many NLP tasks require to automatically identify the most significant words in a text. In this work, we derive word significance from models trained to solve semantic task: Natural Language Inference and Paraphrase Identification. Using an attribution method aimed to explain the predictions of these models, we derive importance scores for each input token. We evaluate their relevance using a so-called cross-task evaluation: Analyzing the performance of one model on an input masked according to the other model's weight, we show that our method is robust with respect to the choice of the initial task. Additionally, we investigate the scores from the syntax point of view and observe interesting patterns, e.g. words closer to the root of a syntactic tree receive higher importance scores. Altogether, these observations suggest that our method can be used to identify important words in sentences without any explicit word importance labeling in training.

1 Introduction

The ability to decide which words in a sentence are semantically important plays a crucial role in various areas of NLP (e.g. compression, paraphrasing, summarization, keyword identification). One way to compute (semantic) word significance for compression purposes is to rely on syntactic patterns, using Integer Linear Programming techniques to combine several sources of information (Clarke and Lapata, 2006; Filippova and Strube, 2008). Xu and Grishman (2009) exploit the same cues, with significance score computed as a mixture of TF-IDF and surface syntactic cues. A similar approach estimates word importance for summarization (Hong and Nenkova, 2014) or learns these significance scores from word embeddings (Schakel and Wilson, 2015; Sheikh et al., 2016).

Significance scores are also useful in an entirely different context, that of explaining the decisions of

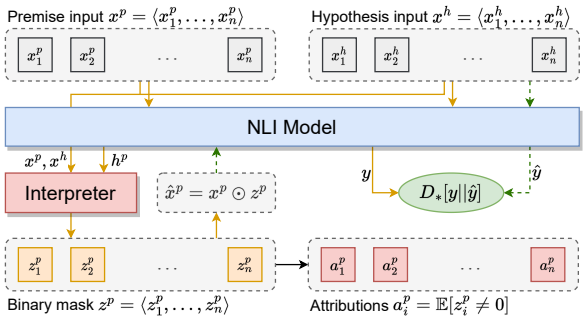

Figure 1: The first pass (yellow plain arrows): A premise and hypothesis are passed to the NLI model. The interpreter takes both text inputs x^p, x^h, and hidden states h^p of the NLI model's encoder. It generates a binary mask z^p which is used to mask x^p, resulting in $\hat{x}^p$. The second pass (green dashed arrows): $\hat{x}^p$ is passed to the NLI model together with the original hypothesis. The divergence D_* minimizes the difference between predicted distributions y and $\hat{y}$ of these two passes.

Deep Neural Networks (DNNs). This includes investigating and interpreting hidden representations via auxiliary probing tasks (Adi et al., 2016; Conneau et al., 2018); quantifying the importance of input words in the decisions computed by DNNs in terms of analyzing attention patterns (Clark et al., 2019); or using attribution methods based on attention (Vashishth et al., 2019), back-propagation (Sundararajan et al., 2017) or perturbation techniques (Guan et al., 2019; Schulz et al., 2020). Along these lines, DeYoung et al. (2020) present a benchmark for evaluating the quality of model-generated rationals compared to human rationals.

In this study, we propose to use such techniques to compute semantic significance scores in an innovative way. We demand the scores to have these intuitive properties: (a) Content words are more important than function words; (b) Scores are context-dependent; (c) Removing low-score words minimally changes the sentence meaning. For this, we train models for two semantic tasks, Natural Lan-

Findings of the Association for Computational Linguistics: ACL 2023, pages 8846–8856
July 9-14, 2023

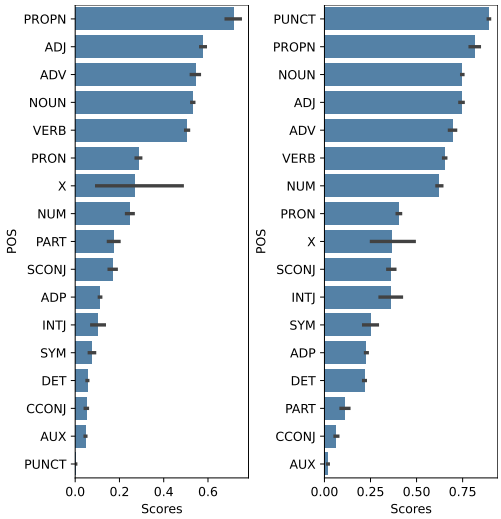

Figure 2: Average scores for each POS category for the NLI model (left) and PI model (right).

guage Inference and Paraphrase Identification, and use the attribution approach of De Cao et al. (2020) to explain the models' predictions. We evaluate the relevance of scores using the so-called *cross-task evaluation*: Analyzing the performance of one model on an input masked according to the other model's weights. We show that our method is robust with respect to the choice of the initial task and fulfills all our requirements. Additionally, hinting at the fact that trained hidden representations encode a substantial amount of linguistic information about morphology (Belinkov et al., 2017), syntax (Clark et al., 2019; Hewitt and Manning, 2019), or both (Peters et al., 2018), we also analyze the correlations of our scores with syntactic patterns.

2 Method

We assume that sentence-level word significance (or word importance) is assessed by the amount of contribution to the overall meaning of the sentence. This means that removing low-scored word should only slightly change the sentence meaning.

The method we explore to compute significance score repurposes attribution techniques originally introduced to explain the predictions of a DNN trained for a specific task. Attribution methods typically compute sentence level scores for each input word, identifying the ones that contribute most to the decision. By explicitly targeting semantic prediction tasks, we hope to extract attribution scores that correlate well with semantic significance.

Our significance scoring procedure thus consists of two main components: an underlying model and an interpreter. The underlying model is trained to solve a semantic task. We select two tasks: Natural Language Inference (NLI) — classifying the relationship of a premise–hypothesis pair into entailment, neutrality or contradiction — and Paraphrase Identification (PI) — determining whether a pair of sentences have the same meaning.

The interpreter relies on the attribution method proposed by De Cao et al. (2020), seeking to mask the largest possible number of words in a sentence, while at the same time preserving the underlying model's decision obtained from the full sentence pair. The interpreter thus minimizes a loss function comprising two terms: an L_0 term, on the one hand, forces the interpreter to maximize the number of masked elements, and a divergence term D_*, on the other hand, aims to diminish the difference between the predictions of the underlying model when given (a) the original input or (b) the masked input.

We take the outputs of the interpreter, i.e. the attribution scores, as probabilities that given words are not masked. Following De Cao et al. (2020), these probabilities are computed assuming an underlying Hard Concrete distribution on the closed interval $[0, 1]$, which assigns a non-zero probability to extreme values (0 and 1) (Fig. 9, De Cao et al., 2020). During interpreter training, a reparametrization trick is used (so that the gradient can be propagated backwards) to estimate its parameters. Given the Hard Concrete distribution output, the attribution score for a token expresses the expectation of sampling a non-zero value, meaning that the token should be masked (Section 2, Stochastic masks, De Cao et al., 2020). We illustrate the process in Figure 1.

3 Experimental Setup

3.1 Underlying Models

We use a custom implementation of a variant of the Transformer architecture (Vaswani et al., 2017) which comprises two encoders sharing their weights, one for each input sentence. This design choice is critical as it allows us to compute importance weights of isolated sentences, which is what we need to do in inference. We then concatenate encoder outputs into one sequence from which a fully connected layer predicts the class, inspired by Sentence-BERT (Reimers and Gurevych, 2019) architecture. See Appendix A.1 for a discussion on the architecture choice, and for datasets, implementation and training details.

3.2 Interpreter

We use the attribution method introduced by De Cao et al. (2020). The interpreter consists of classifiers, each processing hidden states of one layer and predicting the probability whether to keep or discard input tokens. See Appendix A.2 for datasets, implementation and training details.[1]

4 Analysis

In our analysis of the predicted masks, we only consider the last-layer classifier, rescaling the values so that the lowest value and the highest value within one sentence receive the scores of zero and one, respectively. All results use the SNLI validation set.

4.1 Content Words are More Important

We first examine the scores that are assigned to content and functional words. We compute the average score for each POS tag (Zeman et al., 2022) and display the results in Figure 2. For both models, Proper Nouns, Nouns, Pronouns, Verbs, Adjectives and Adverbs have leading scores. Determiners, Particles, Symbols, Conjunctions, Adpositions are scored lower. We observe an inconsistency of the PI model scores for Punctuation. We suppose this reflects idiosyncrasies of the PI dataset: Some items contain two sentences within one segment, and these form a paraphrase pair only when the other segment also consists of two sentences. Therefore, the PI model is more sensitive to Punctuation than expected. We also notice the estimated importance of the X category varies widely, which is expected since this category is, based on its definition, a mixture of diverse word types. Overall, these results fulfil our requirement that content words achieve higher scores than function words.

4.2 Word Significance is Context-Dependent

We then question the ability of the interpreter to generate context-dependent attributions, contrasting with purely lexical measures such as TF-IDF. To answer this question, we compute the distribution of differences between the lowest and highest scores for words having at least 100 occurrences in the training and 10 in the validation data, excluding tokens containing special characters or numerals. The full distribution is plotted in Figure 3.

Scores extracted from both models report increased distribution density towards larger differences, confirming that significance scores are not lexicalized, but instead strongly vary according to the context for the majority of words. The greatest difference in scores for PI model is around 0.5, the analysis of the NLI model brings this difference even more towards 1. We explain it by the nature of datasets: It is more likely that the NLI model's decision relies mostly on one or on a small group of words, especially in the case of contradictions.

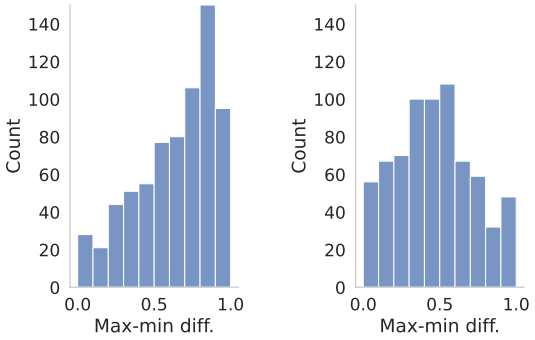

Figure 3: The NLI model (left), PI model (right) and the distribution of differences between the maximal and minimal value for each token.

4.3 Cross-Task Evaluation

In this section, we address the validity of importance scores. We evaluate the models using so-called *cross-task evaluation*: For model A, we take its validation dataset and gradually remove a portion of the lowest scored tokens according to the interpreter of model B. We then collect the predictions of model A using the malformed inputs and compare it to a baseline where we randomly remove the same number of tokens. We evaluate both models in this setting, however, since the results for both models have similar properties, we report here only the analysis of the PI model in Table 1. See Appendix B for the NLI model results.

Table 1 reports large differences in performance when the tokens are removed according to our scores, compared to random removal. When one third of tokens from both sentences is discarded, the PI model performance decreases by 2.5%, whereas a random removal causes a 15.1% drop (Table 1, 4th row and 4th column). The models differ most when a half of the tokens are removed, resulting in a difference in accuracy of 18.3% compared to the baseline (Table 1, 6th row and 6th column). Examining performance up to the removal of 20% of tokens, the difference between the random and

[1] Our source code with the license specification is available at https://github.com/J4VORSKY/word-importance

	PI Model performance										
	0%	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
0%	*85.1↑0.0*	*84.7↑0.7*	84.5↑4.6	83.0↑6.8	80.9↑9.1	77.7↑12.2	74.3↑12.9	69.3↑10.6	62.6↑7.3	56.0↑4.0	*50.0↑0.0*
10%	*84.7↑0.9*	84.7↑2.0	84.4↑5.7	82.8↑7.6	81.0↑9.9	77.8↑12.9	74.5↑13.4	69.5↑11.3	62.6↑7.5	55.8↑3.8	*50.0↑0.0*
20%	84.2↑4.1	84.2↑5.2	84.3↑8.3	83.0↑10.3	81.5↑12.2	78.4↑14.7	74.9↑14.4	70.1↑12.3	63.0↑8.2	56.2↑4.3	*50.0↑0.0*
30%	83.1↑6.9	83.1↑7.7	83.3↑11.0	82.6↑12.6	81.8↑15.0	79.0↑16.1	75.6↑15.7	70.9↑13.3	63.5↑8.6	56.3↑4.6	*50.0↑0.1*
40%	80.7↑9.9	80.4↑10.4	81.0↑12.7	81.0↑14.0	80.9↑16.1	78.7↑17.9	75.5↑16.1	71.1↑13.7	64.2↑9.9	56.7↑5.0	*50.0↑0.1*
50%	77.3↑11.3	77.5↑11.6	78.1↑13.5	78.9↑15.0	78.8↑16.6	78.0↑18.3	75.2↑17.0	71.2↑15.0	64.2↑9.6	56.8↑5.0	*50.0↑0.1*
60%	73.6↑11.7	73.9↑12.0	74.4↑13.3	75.9↑15.2	75.3↑16.4	75.9↑17.9	74.4↑17.4	71.2↑15.7	65.3↑11.2	57.1↑5.2	*49.9↓0.2*
70%	68.4↑10.3	68.8↑11.1	68.7↑11.3	70.2↑12.8	70.7↑14.3	71.1↑15.3	71.0↑15.9	70.3↑15.4	66.4↑13.3	58.2↑6.0	*50.0↓0.3*
80%	62.3↑7.3	62.3↑7.5	62.4↑7.6	63.2↑8.7	63.6↑9.3	64.3↑10.4	64.7↑11.1	65.8↑12.6	67.0↑15.0	59.8↑8.2	*49.7↓0.4*
90%	56.2↑4.0	56.3↑4.1	56.5↑4.4	56.7↑4.7	57.2↑5.3	57.2↑5.4	57.5↑5.5	58.5↑7.1	60.5↑8.8	63.9↑12.1	50.2↓2.4
100%	*50.0↑0.0*	*50.0↓0.0*	*50.0↑0.0*	*50.0↑0.1*	*50.0↑0.2*	*50.1↑0.1*	*50.0↑0.1*	*50.0↓0.1*	*50.1↓0.2*	*50.5↓0.5*	*50.0↑0.0*

Table 1: The accuracy of the PI model when a given percentage of the least important input tokens are removed from the first sentence (rows) or the second (columns) according to the NLI model's weights. Each cell contains the model accuracy (left), difference in comparison to the randomized baseline model (right) and an arrow denoting the increase (↑) or decrease (↓) in performance of our model compared to the baseline. The difference of values in *italics* is *not* statistically significant ($p < 0.01$).

	NLI Model		PI Model		
Depth	**Avg**	**Std**	**Avg**	**Std**	**Count**
1	**0.52**	0.35	**0.64**	0.31	9424
2	**0.36**	0.36	**0.53**	0.39	27330
3	**0.23**	0.31	**0.40**	0.35	26331
4	0.22	0.31	0.33	0.36	7183
5	0.22	0.30	0.35	0.35	1816

Table 2: Importance scores of tokens for each depth in syntactic trees. Stat. significant differences between the current and next row are bolded ($p < 0.01$).

	NLI Model		PI Model		
Dependency Relation	**Avg**	**Std**	**Avg**	**Std**	**Count**
det, case, cop, cc, punct, mark	-0.50	0.37	-0.37	0.49	34034
advcl, acl, xcomp	0.11	0.43	0.06	0.38	2789
nsubj	-0.22	0.45	0.06	0.39	9323
punct	-0.53	0.35	0.24	0.35	8148
compound	0.07	0.46	-0.04	0.35	2437

Table 3: The average and standard deviation of significance scores, and the count of aggregated dependency relations in syntactic trees.

importance-based word removal are not so significant, probably because of the inherent robustness of the PI model which mitigates the effect of the (random) removal of some important tokens. On the other hand, removing half of the tokens is bound to have strong effects on the accuracy of the PI model, especially when some important words are removed (in the random deletion scheme); this is where removing words based on their low importance score makes the largest difference. At higher dropping rates, the random and the importance-based method tend to remove increasing portions of similar words, and their scores tend to converge (in the limiting case of 100% removal, both strategies have exactly the same effect). Overall, these results confirm that our method is robust with respect to the choice of the initial task and that it delivers scores that actually reflect word importance.

4.4 Important Words are High in the Tree

Linguistic theories differ in ways of defining dependency relations between words. One established approach is motivated by the 'reducibility' of sentences (Lopatková et al., 2005), i.e. gradual removal of words while preserving the grammatical correctness of the sentence. In this section, we study how such relationships are also observable in attributions. We collected syntactic trees of input sentences with UDPipe (Straka, 2018),[2] which reflect syntactic properties of the UD format (Zeman et al., 2022).[3] When processing the trees, we discard punctuation and compute the average score of all tokens for every depth level in the syntactic trees. We display the first 5 depth levels in Table 2.

We can see tokens closer to the root in the syntactic tree obtain higher scores on average. We measure the correlation between scores and tree levels, resulting in -0.31 Spearman coefficient for the NLI model and -0.24 for the PI model. Negative coefficients correctly reflect the tendency of the scores to decrease in lower tree levels. It thus appears that attributions are well correlated with word positions in syntactic trees, revealing a relationship between semantic importance and syntactic position.

4.5 Dependency Relations

We additionally analyze dependency relations occurring more than 100 times by computing the

[2] https://lindat.mff.cuni.cz/services/udpipe/

[3] UD favors relations between content words, function words are systematically leaves in the tree. However, having function words as leaves better matches our perspective of information importance flow, unlike in Gerdes et al. (2018).

score difference between child and parent nodes, and averaging them for each dependency type. In Table 3, we depict relations which have noteworthy properties with respect to significance scores (the full picture is in Appendix C). Negative scores denote a decrease of word significance from a parent to its child. We make the following observations.

The first row of the table illustrates dependencies that have no or very limited contribution to the overall meaning of the sentence. Looking at the corresponding importance scores, we observe that they are consistently negative, which is in line with our understanding of these dependencies.

The second row corresponds to cases of clausal relationships. We see an increase in importance scores. This can be explained since the dependents in these relationships are often heads of a clause, and thus contribute, probably more than their governor, to the sentence meaning. It shows models' ability to detect some deep syntactic connections.

The last block represents relations that are not consistent across the models. Nominal Subject is judged less important in the NLI model than in the PI model. As mentioned in Section 4.1, Punctuation differs similarly. Elements of Compound are preferred in different orders depending on the model. On the other hand, all other relation types are consistent: Ranking each type of dependency relation based on its average score and calculating correlation across our models results in 0.73 Spearman coefficient. This reveals a strong correlation between importance and syntactic roles.

5 Conclusion

In this paper, we have proposed a novel method to compute word importance scores using attribution methods, aiming to explain the decisions of models trained for semantic tasks. We have shown these scores have desired and meaningful properties: Content words are more important, scores are context-dependent and robust with respect to the underlying semantic task. In our future work, we intend to exploit these word importance scores in various downstream applications.

Limitations

Our method of identifying important words requires a dataset for a semantic task (in our case NLI or PI), which limits its applicability. This requirement also prevents us from generalizing our observations too broadly: we tested our method only on one high-resource language where both dependency parsers and NLI / PI datasets are available. Our analysis also lacks the comparison to other indicators of word significance.

Acknowledgements

The work has been partially supported by the grants 272323 of the Grant Agency of Charles University, 19-26934X (NEUREM3) of the Czech Science Foundation and SVV project number 260 698. A part of this work has been done at Laboratoire Interdisciplinaire des Sciences du Numérique (LISN) in Orsay, France.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

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 the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

James Clarke and Mirella Lapata. 2006. Constraint-based sentence compression: An integer programming approach. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 144–151, Sydney, Australia. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. 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)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Nicola De Cao, Michael Sejr Schlichtkrull, Wilker Aziz, and Ivan Titov. 2020. How do decisions emerge across layers in neural models? interpretation with differentiable masking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3243–3255, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 25–32, Salt Fork, Ohio, USA. Association for Computational Linguistics.

Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2018. SUD or surface-syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 66–74, Brussels, Belgium. Association for Computational Linguistics.

Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. 2019. Towards a deep and unified understanding of deep neural models in NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2454–2463. PMLR.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 712–721, Gothenburg, Sweden. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic gradient descent. In *ICLR: International Conference on Learning Representations*, pages 1–15.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Markéta Lopatková, Martin Plátek, and Vladislav Kuboň. 2005. Modeling syntax of free word-order languages: Dependency analysis by reduction. In *International Conference on Text, Speech and Dialogue*, pages 140–147. Springer.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Adriaan MJ Schakel and Benjamin J Wilson. 2015. Measuring word significance using distributed representations of words. *arXiv preprint arXiv:1508.02297*.

Karl Schulz, Leon Sixt, Federico Tombari, and Tim Landgraf. 2020. Restricting the flow: Information bottlenecks for attribution. In *International Conference on Learning Representations*.

Imran Sheikh, Irina Illina, Dominique Fohr, and Georges Linarès. 2016. Learning word importance with the neural bag-of-words model. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 222–229, Berlin, Germany. Association for Computational Linguistics.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across NLP tasks. *arXiv preprint arXiv:1909.11218*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Wei Xu and Ralph Grishman. 2009. A parse-and-trim approach with information significance for Chinese sentence compression. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation (UCNLG+Sum 2009)*, pages 48–55, Suntec, Singapore. Association for Computational Linguistics.

Daniel Zeman et al. 2022. Universal dependencies 2.10. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

A Training

A.1 Underlying Models

Implementation Language modeling often treats the input of semantic classification tasks as a one-sequence input, even for tasks involving multiple sentences on the input side (Devlin et al., 2019; Lewis et al., 2020; Lan et al., 2020). However, processing two sentences as one irremediably compounds their hidden representations. As we wish to separate representations of single sentences, we resort to a custom implementation based on the Transformers architecture (Vaswani et al., 2017), which comprises two encoders (6 layers, 8 att. heads, 1024 feed forward net. size, 512 emb. size) sharing their weights, one for each input sentence. Following Sentence-BERT (Reimers and Gurevych, 2019), we computed the mean of the encoder output sentence representations u and v, and concatenated them to an additional $|u - v|$ term. This was passed to a linear layer for performing the final classification. We implemented models in `fairseq` (Ott et al., 2019).[4]

Datasets The NLI model was trained on SNLI (Bowman et al., 2015)[5], MULTI_NLI (Williams et al., 2018)[6] and QNLI (Rajpurkar et al., 2016)[7] datasets. Since QNLI uses a binary scheme ('entailment' or 'non-entailment'), we interpret 'non-entailment' as a neutral relationship. Table 6 describes the NLI training and validation data. The PI model was trained on QUORA Question Pairs[8] and PAWS (Zhang et al., 2019)[9] datasets. We swapped a random half of sentences in the data to ensure the equivalence of both sides of the data. Table 7 displays the PI training and validating data.

Training We trained both models using an adaptive learning rate optimizer ($\alpha = 3 \times 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.98$) (Kingma and Ba, 2015) and a inverse square root scheduler with 500 warm-up updates. We trained with 64k maximum batch tokens over 6 epochs with 0.1 dropout regulation. We trained on an NVIDIA A40 GPU using half-precision floating-point format FP16, which took less than 2 hours for both models. The PI model and NLI model achieve 85.1% and 78.4% accuracy

[4] https://github.com/facebookresearch/fairseq
[5] https://huggingface.co/datasets/snli
[6] https://huggingface.co/datasets/multi_nli
[7] https://huggingface.co/datasets/glue#qnli
[8] https://huggingface.co/datasets/quora
[9] https://huggingface.co/datasets/paws

	NLI Model performance										
	0%	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
0%	*78.4↑0.0*	*78.1↑0.5*	77.9↑3.1	77.1↑5.5	75.8↑8.1	72.0↑8.9	68.6↑8.5	63.7↑7.3	55.9↑5.1	46.8↑3.1	*33.5↑0.0*
10%	*78.4↑1.4*	78.3↑1.8	78.1↑4.7	77.2↑6.5	75.7↑8.9	72.1↑9.4	68.6↑9.0	63.6↑7.6	55.8↑5.5	46.6↑3.0	*33.6↑0.1*
20%	78.0↑4.1	77.8↑4.3	77.7↑6.4	77.1↑8.4	75.4↑9.7	72.0↑10.2	68.2↑9.6	63.6↑8.3	55.7↑5.7	46.7↑3.8	*33.5↑0.3*
30%	77.3↑6.5	77.2↑6.6	77.0↑8.8	76.7↑10.3	74.9↑11.2	71.3↑11.8	68.1↑11.1	63.2↑8.9	55.7↑6.7	46.6↑3.9	*33.4↑0.6*
40%	76.1↑8.3	76.0↑8.6	75.9↑9.9	75.3↑11.0	74.0↑11.9	71.1↑12.5	67.4↑10.9	63.1↑9.5	55.7↑7.7	47.1↑5.1	*33.5↑0.2*
50%	72.8↑8.6	72.7↑8.6	73.1↑10.2	72.4↑10.2	71.5↑11.3	69.3↑12.6	66.7↑12.4	62.4↑10.2	55.5↑8.1	46.4↑4.5	*33.5↓0.2*
60%	68.7↑6.7	68.5↑6.9	68.9↑7.9	68.6↑9.1	67.7↑9.5	66.1↑10.6	64.3↑10.8	60.8↑9.9	54.0↑6.9	45.9↑3.8	*33.4↓0.2*
70%	63.2↑5.3	63.0↑5.2	63.5↑6.3	62.9↑6.0	62.2↑7.0	61.3↑7.8	60.2↑8.8	58.1↑9.5	52.5↑6.2	45.1↑3.4	*33.4↑0.1*
80%	57.4↑3.6	57.3↑3.6	57.7↑3.7	57.2↑3.3	57.1↑4.1	56.5↑5.4	55.1↑4.9	53.8↑6.0	50.3↑4.9	44.9↑3.7	*33.4↓0.0*
90%	52.5↑2.1	52.4↑2.1	52.9↑2.6	52.8↑2.1	52.4↑2.3	51.9↑2.9	51.2↑2.7	49.9↑2.5	47.6↑3.2	43.5↑3.2	*33.7↑0.4*
100%	*42.8↑0.0*	*42.8↑0.1*	*43.5↑0.1*	*43.8↑0.2*	*44.5↑0.5*	*44.7↑0.5*	*45.1↑0.4*	*44.2↓0.8*	*43.1↓0.1*	*40.2↑0.3*	*33.8↑0.0*

Table 4: The accuracy of the NLI model when a given percentage of the least important input tokens are removed from the premise (rows) or hypothesis (columns) according to the PI model's weights. The description of the cell content is the same as in Table 3.

	NLI Model		PI Model			
Dep. Rel.	**Avg**	**Std**	**Avg**	**Std**	**Count**	**Description**
cop	-0.74	0.30	-0.74	0.27	1623	Copula, e.g. John *is* the best dancer; Bill *is* honest
case	-0.55	0.35	-0.54	0.30	7651	Case Marking, e.g. the Chair *'s* office; the office *of* the Chair
punct	-0.53	0.35	0.24	0.35	8148	Punctuation, e.g. Go home *!*
aux	-0.51	0.34	-0.67	0.27	4622	Auxiliary, e.g. John *has* died; he *should* leave
cc	-0.48	0.32	-0.74	0.23	707	Coordinating Conjunction, e.g. *and* yellow
det	-0.45	0.38	-0.55	0.38	14801	Determiner, e.g. *the* man
mark	-0.39	0.34	-0.48	0.31	1104	Marker, e.g. *before*; *after*; *with*; *without*
nsubj	-0.22	0.45	0.06	0.39	9323	Nominal Subject, e.g. *John* won
nummod	-0.10	0.37	-0.02	0.38	1269	Numeric Modifier, e.g. *forty* dollars, *3* sheep
nmod	-0.06	0.52	-0.13	0.42	3153	Nominal Modifier, e.g. the office of the *Chair*
advmod	-0.01	0.51	-0.01	0.41	1299	Adverbial Modifier, e.g. *genetically* modified, *less* often
advcl	0.05	0.43	0.05	0.33	857	Adverbial Clause Modifier, e.g. if you know who did it, you should *say* it
compound	0.07	0.46	-0.04	0.35	2437	Compound, e.g. *phone* book; *ice* cream
conj	0.10	0.41	0.03	0.28	742	Conjunct, e.g. big and *yellow*
acl	0.11	0.43	0.04	0.41	1367	Adnominal Clause), e.g. the issues as he *sees* them; a simple way to *get*
amod	0.11	0.42	-0.01	0.32	2974	Adjectival Modifier, e.g. *big* boat
obl	0.16	0.47	0.09	0.33	5002	Oblique Nominal, e.g. last *night*, I swam in the *pool*
xcomp	0.21	0.41	0.12	0.38	565	Open Clausal Complement, e.g. I started to *work*
obj	0.25	0.44	0.12	0.36	4377	Object, e.g. she got a *gift*

Table 5: The average and standard deviation of significance scores, and the count and a short description of each dependency relation between a parent and *child* node in the syntactic tree.

	Training			
	Entail.	**Neutral.**	**Contra.**	**All**
SNLI	183k	183k	183k	549k
QNLI	52k	52k	-	105k
MULTI_NLI	131k	131k	131k	393k
All	366k	366k	315k	1047k

	Validating			
	Entail.	**Neutral.**	**Contra.**	**All**
SNLI	3.3k	3.3k	3.3k	10k

Table 6: The number of samples in training (top) and validation (bottom) data for the NLI model.

	Training		
	Paraphrase	**Non Paraphrase**	**All**
QUORA	146k	248k	394k
PAWS	25k	55k	80k
All	171k	303k	474k

	Validating		
	Paraphrase	**Non Paraphrase**	**All**
QUORA	3.4k	3.4k	6.8k

Table 7: The number of samples in training (top) and validation (bottom) data for the PI model.

on corresponding validating sets, respectively. We consider this performance sufficient given limitations put on the architecture choice.

A.2 Interpreter

Implementation We use the attribution method introduced by De Cao et al. (2020). Assuming L layers for the NLI encoder, the interpreter model contains $L+1$ classifiers. Each classifier is a single-hidden-layer MLP, which inputs hidden states and predicts binary probabilities whether to keep or discard input tokens. The implementation details closely follow the original work.

Training We trained on the first 50k samples of the corresponding underlying model's training data, using a learning rate $\alpha = 3 \times 10^{-5}$ and a divergence constrain $D_* < 0.1$. The number of training samples and the rest of hyper-parameters

follow the original work. We trained over 4 epochs with a batch size of 64.

B Cross-Task Evaluation

The performance of the NLI model in the cross-task evaluation, compared to the baseline model, is displayed in Table 4.

C Dependency Relations

We examined all dependency relations with a frequency greater than 100 by computing the score difference between child and parent nodes, and averaging them for each every dependency type. Results are in Table 5.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
After Conclusion

☒ A2. Did you discuss any potential risks of your work?
We believe that our work has no potential risks

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Appendix A

☑ B1. Did you cite the creators of artifacts you used?
Appendix A

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
3.2

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
3.2

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
We do not publish any data and the data we use are publicly available and used in several studies

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Appendix A

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix A

C ☑ Did you run computational experiments?

Appendix A

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.