Assessing Word Importance Using Models Trained for Semantic Tasks
Dávid Javorský, Ondřej Bojar, François Yvon

To cite this version:

HAL Id: hal-04163044
https://hal.science/hal-04163044
Submitted on 17 Jul 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - ShareAlike 4.0 International License
Assessing Word Importance Using Models Trained for Semantic Tasks

Dávid Javorský1 and Ondřej Bojar1 and François Yvon2

1Charles University, Faculty of Mathematics and Physics, Prague, Czechia
2Sorbonne 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 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-
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 words should only slightly change the sentence meaning.

The method we explore to compute significance scores 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.

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 words should only slightly change the sentence meaning.

The method we explore to compute significance scores 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.\footnote{Our source code with the license specification is available at https://github.com/J4VORSKY/word-importance}

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 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 fulfill 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.

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
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).

<table>
<thead>
<tr>
<th>Depth</th>
<th>NLI Model</th>
<th>PI Model</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.52</td>
<td>0.35</td>
<td>9424</td>
</tr>
<tr>
<td>2</td>
<td>0.36</td>
<td>0.39</td>
<td>27330</td>
</tr>
<tr>
<td>3</td>
<td>0.23</td>
<td>0.35</td>
<td>26331</td>
</tr>
<tr>
<td>4</td>
<td>0.22</td>
<td>0.35</td>
<td>7183</td>
</tr>
<tr>
<td>5</td>
<td>0.22</td>
<td>0.35</td>
<td>1816</td>
</tr>
</tbody>
</table>

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).

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),

which reflect syntactic properties of the UD format (Zeman et al., 2022). 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

\[\text{https://lindat.mff.cuni.cz/services/udpipe/}\]

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


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 $v$ and $v^\beta$, and concatenated them to an additional $|v − v^\beta|$ term. This was passed to a linear layer for performing the final classification. We implemented models in fairseq (Ott et al., 2019).

Datasets The NLI model was trained on SNLI (Bowman et al., 2015), MULTILINGUAL (Williams et al., 2018) and QNLI (Rajpurkar et al., 2016) 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 and PAWS (Zhang et al., 2019) 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 respectively.
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.

<table>
<thead>
<tr>
<th>NLI Model</th>
<th>PI Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dep. Rel.</td>
<td>Avg</td>
</tr>
<tr>
<td>cop</td>
<td>0.14</td>
</tr>
<tr>
<td>case</td>
<td>-0.55</td>
</tr>
<tr>
<td>punct</td>
<td>-0.53</td>
</tr>
<tr>
<td>aux</td>
<td>-0.51</td>
</tr>
<tr>
<td>cc</td>
<td>-0.48</td>
</tr>
<tr>
<td>det</td>
<td>-0.45</td>
</tr>
<tr>
<td>mark</td>
<td>-0.39</td>
</tr>
<tr>
<td>nsubj</td>
<td>-0.22</td>
</tr>
<tr>
<td>nummod</td>
<td>-0.10</td>
</tr>
<tr>
<td>advmod</td>
<td>-0.06</td>
</tr>
<tr>
<td>advc1</td>
<td>0.05</td>
</tr>
<tr>
<td>compound</td>
<td>0.07</td>
</tr>
<tr>
<td>conj</td>
<td>0.10</td>
</tr>
<tr>
<td>acl</td>
<td>0.11</td>
</tr>
<tr>
<td>amod</td>
<td>0.11</td>
</tr>
<tr>
<td>obl</td>
<td>0.16</td>
</tr>
<tr>
<td>xcomp</td>
<td>0.21</td>
</tr>
<tr>
<td>obj</td>
<td>0.25</td>
</tr>
</tbody>
</table>

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.

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

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 $D_{\ast} < 0.1$. The number of training samples and the rest of hyper-parameters are:

<table>
<thead>
<tr>
<th>Hyper-parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch Size</td>
<td>128</td>
</tr>
<tr>
<td>Dropout Rate</td>
<td>0.2</td>
</tr>
<tr>
<td>Learning Rate</td>
<td>1e-5</td>
</tr>
<tr>
<td>Weight Decay</td>
<td>0.01</td>
</tr>
<tr>
<td>Momentum</td>
<td>0.9</td>
</tr>
<tr>
<td>Max Epochs</td>
<td>5000</td>
</tr>
<tr>
<td>Early Stopping</td>
<td>500</td>
</tr>
<tr>
<td>Patience</td>
<td>10000</td>
</tr>
</tbody>
</table>

8853
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?
   Yes

✗ 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?
   Yes

✓ 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)?
   Yes

✗ 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  X 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.