A sad study on what is inside a Neural Language Model

SLLD Modules 1 & 2

Irene Dini

May 19, 2023

Abstract

Over the years, the introduction of Neural Language Models (NLMs) has revolutionized the field of Natural Language Processing (NLP). With the advent of NLMs, we have moved away from encoding text using explicit and understandable features, towards providing raw text to the model and letting it encode it into a set of implicit features. This approach has led to significantly improved performance, but it is not clear what information is being considered relevant. The aim of this project is to investigate the implicit representations of a NLM to understand whether it encodes linguistic information and where it does so.

1 Introduction

Natural language processing (NLP) is a field of computer science and linguistics that deals with the development of algorithms to solve problems that require the analysis of natural language. In the early stages of research, NLP systems were classifiers (for example, based on Support Vector Machines or Recurrent Neural Networks) trained to solve specific tasks. These classifiers received text encoded into manually engineered features capturing syntactic and semantic information (Figure 1a). However, recent advances in machine learning have revolutionized the way NLP problems are solved. Current models, called Neural Language Models (NLMs), are based on deep neural networks trained on natural language to learn to extract syntactic and semantic knowledge from text, autonomously defining a set of interesting features (Figure 1b). By analyzing massive amounts of text, these models learn to create a compressed and numerical representation of the text that capture important characteristics. Only after this preliminary training phase (called pre-training) the models are further trained to solve a specific task via the so-called fine-tuning phase. These models outperform earlier systems and humans in many tasks obtaining better results and fast predictions. However, this improvement comes at the cost of interpretability since they rely on complex neural networks that offer little transparency about their inner workings and abilities.

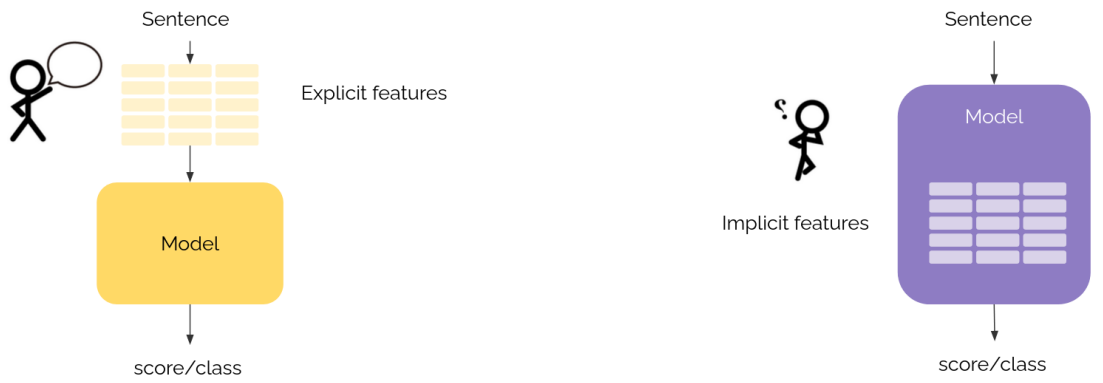
1.1 Neural Language Models

As mentioned in the introduction, in recent years, there has been a shift from traditional NLP approaches to approaches based on the use of Neural Language Models. Language modeling is the task of assigning a probability score to a sequence of words. It can also be seen as the task of predicting the next word in a sequence of words, given the previous words in that sequence. An example of language modeling performed on a sentence can be found in Figure 2. Given the sequence "*Can you please come*" the language model predicts the word "*here*".

Neural language models, are language models based on (deep) neural networks. They are trained with the language modeling task over a large collection of textual data, to learn important statistics on a language, that will be the base for the resolution of a target task like sentiment-analysis, named entity extraction, topic classification, etc. learned in a second training phase, called fine-tuning.

1.2 The transformer model

In 2017, with the publication of "Attention is all you need" ([6]) the NLP world underwent a revolution. This paper introduces the Transformer model: an encoder-decoder architecture that employs the



(a) Traditional pipeline

(b) NLMs pipeline

Figure 1: Difference between the traditional NLP pipeline where features where hand-defined, and modern pipeline where the features are automatically created by Neural Language Models.

Figure 2: Language modeling task

attention mechanism to to draw global dependencies between input and output. In general, encoder-decoder models encode an input sequence of symbol representations $x = (x_1, ..., x_n)$ to a sequence of continuous representations $z = (z_1, ..., z_n)$. Then the decoder generates an output sequence $y = (y_1, ..., y_m)$ one element at a time, given z and the last generated symbol. The Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and the decoder, as shown in Figure 3a. Part of the transformer layer is the attention head: the attention function can be viewed as mapping a query and a set of key-value pairs to an output, where the query, keys, values and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. Formally speaking, given the query matrix Q, the key matrix K and the value matrix V as inputs, the output is computed as:

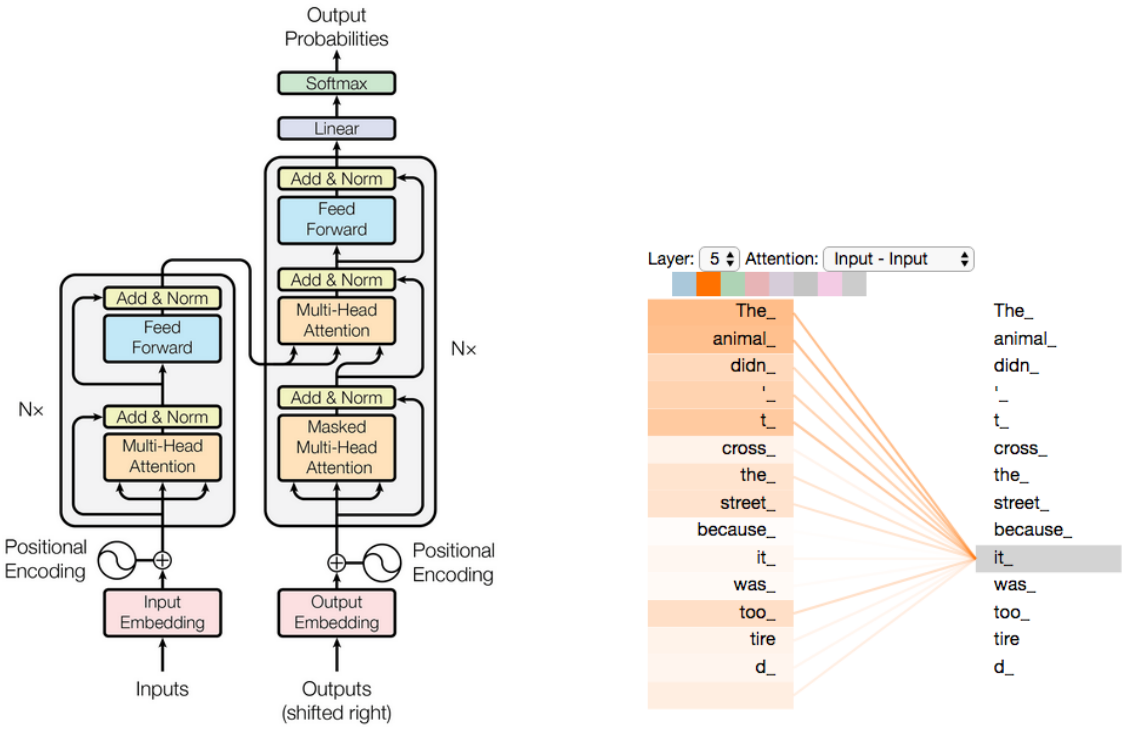
$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where dk is the dimension of the query matrix. As the model processes each word (each position in the input sequence), self attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word. A graphical visualization of attention is illustrated in figure 3b.

Although originally proposed to solve the problem of machine translation, given its ability of better modeling long-term dependencies, the Transformer model was successfully exploited in several works to build highly performative language models like BERT ([2]) and GPT ([4]).

1.3 The BERT model

One of the most popular NLM based on the Transformer architecture is BERT (Bidirectional Encoder Representations from Transformers, [2]). The BERT base version consist of 12 encoders with 12 bidirectional self-attention heads totaling 110 million parameters and was pre-trained on the Toronto BooksCorpus (800 million words) and English Wikipedia (2,500 million words). Differently from the



(a) Transformer

(b) Self-attention

original Transformer model, BERT consists only of the encoder module and is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context. However, since standard conditional language models can only be trained left-to-right or right-to-left (bidirectional conditioning would allow each word to indirectly "see itself"), the authors proposed a a modified version of the original LM task: Masked Language Modeling (MLM). The MLM task consists in masking some percentage of the input tokens at random, and then asking the model to predict those tokens. BERT is also pre-trained with a next sentence prediction task (NSP), i.e. the task of predicting if two sentences are consequent or not. BERT became rapidly a milestone work in the field on NLP, achieving significant empirical results on several tasks, including SQUAD ([5]) and GLUE ([7]). Given its success, several variants of the original model for better language representations have been proposed, such as RoBERTa ([3]) and XLNet ([8]). Moreover, since the vast majority of works focused on how BERT and BERT-based models works on different NLP tasks and less on its workings, in the last years there has been a growing interest in the field of study concerned with investigating the inner behaviour of these models.

2 Experimental Framework

The main goal of this project is to explore the implicit features extracted from a Neural Language Model looking for linguistic information. Specifically, I used a regression model with feature importance calculation to investigate whether there is a small subset of implicit features that contains a significant amount of linguistic information: I used a regression model to predict linguistic features based on these implicit features, and then used feature importance calculations to identify the subset of implicit features that are most predictive of linguistic features.

2.1 Data

The dataset employed in this study comprises 1200 English sentences that have been annotated with complexity scores. These scores, ranging from 1 (indicating very easy comprehension) to 6 (reflecting significant difficulty in comprehension), were determined based on the average rating of perceived complexity from 20 human annotators. The rationale behind selecting this dataset stems from the

strong correlation observed between sentence complexity and various linguistic features. Achieving good performance on the complexity regression task implies a proficient ability to extract linguistic information.

2.2 Feature Extraction

During the experiment I used two different set of features:

- Explicit features: hand-engineered linguistic features, where the meaning of each feature is known;
- Implicit features: a set of features that a language model learned to extract during its pre-training phase.

To ensure consistency, both sets of features underwent standardization before subsequent analyses were conducted.

2.2.1 Explicit features

Regarding the set of explicit features, they were derived utilizing the Profiling-UD tool ([1]), a web-based application specifically designed for linguistic profiling of text, including extensive collections of texts, across multiple languages. This tool enables the extraction of over 130 features, encompassing diverse levels of linguistic description. Examples of these features include sentence length, average number of characters per word, distribution of adjectives, distribution of nouns, distribution of verbs, distribution of present tense verbs, and average syntactic tree length. Given the fact that if a feature is never measured in the dataset, it is discarded from the set of descriptors, for these experiments I had a set of 121 linguistic features. Almost all of these features are continuous, with the exception of a very small number of "counting" features, with values that can vary over very wide ranges (such as the number of words in the sentence) and have therefore been treated as continuous.

2.2.2 Implicit features

As stated before, when presented with a sentence, NLMs generate a numerical representations for each word (alongside a distinct token signifying the sentence itself, the `[CLS]`) using the attention mechanism. Is *this* encoding of words that will be used to create the sentence representation, interpreting each component of the representation vector as an implicit feature. To build the sentence representation starting from words representation, one can either employ the `[CLS]` token's representation or calculate the average of all other word representations. The latter approach generally proves more effective in retaining information, and it is the one employed in this study. When utilizing a neural language model with n layers, is possible to extract n distinct sets of implicit features each sharing the same dimensions, one for each layer. The dimension of the feature space is given by a dimension parameter of the model, called `hidden_size`. The construction of the implicit feature matrices (one for each model's layer) was done by feeding each sentence of the dataset to the model, extracting its representation from each layer, and then stacking together all the sentences representation extracted from the same layer. In this manner, each component of the sentence representation serves as an implicit feature, capturing distinct aspects and characteristics of the input sentences.

2.3 Models

The specific model under investigation in this study is `BERT-tiny`, which represents the smallest available version within the BERT architecture. This particular model consists of 2 layers, each with 2 attention heads, and a hidden dimension of 128. The rationale behind selecting this model is twofold. Firstly, it can be efficiently executed on a CPU, allowing for practical implementation. Secondly, the number of implicit features closely aligns with the number of Profiling-UD features, and it is approximately one order of magnitude lower than the dataset size. Nevertheless, I also conducted parallel experiments employing two larger models: `BERT-small` (consisting of 4 layers, 4 attention heads, and a hidden size of 256) and `BERT-medium` (comprising 8 layers, 8 attention heads, and a hidden size of 512). These additional experiments aimed to examine patterns emerging as the model

size increases. However, it is important to note that the dataset size may not be sufficient to adequately describe the data within such a large implicit feature space.

3 Methodology

The methodology used in this study involved utilizing implicit features to represent the dataset, with each explicit feature treated as a separate target for prediction. Specifically, for each layer of the NLM, I created a matrix of implicit representations of the sentences and trained 121 regressors, one for each Profiling-UD feature. The LASSO regressor was employed as the regression model with regularization coefficients determined using a 5-fold cross-validation. After training the 121 regressors, I obtained 121 feature importance vectors, which indicate the implicit features that were most influential in predicting the explicit features. These feature importance vectors were aggregated into a single feature importance vector using three different methods:

- mean: for each feature, the global importance score is computed as the average of the 121 importance scores.
- max: for each feature, the global importance score is taken as the maximum value of importance obtained in the 121 regressions.
- not zero: for each feature, the global importance score is computed as the number of regressions where the feature was not zeroed.

Once these three global feature importance arrays were obtained, any feature that exhibited outlier behavior in at least one of the arrays was identified as one of the most important. An outlier feature was defined as having an importance value exceeding 3 standard deviations above the average importance.

Subsequently, I examined the performance on the sentence complexity task by training a linear regressor with and without these identified outliers as features. The hypothesis was that if these features indeed encapsulated the majority of the linguistic information, their removal from the training set would significantly diminish the regressor's performance. Conversely, if the regressor was trained using only one feature, using an outlier would yield considerably better performance compared to using a random feature.

4 Results

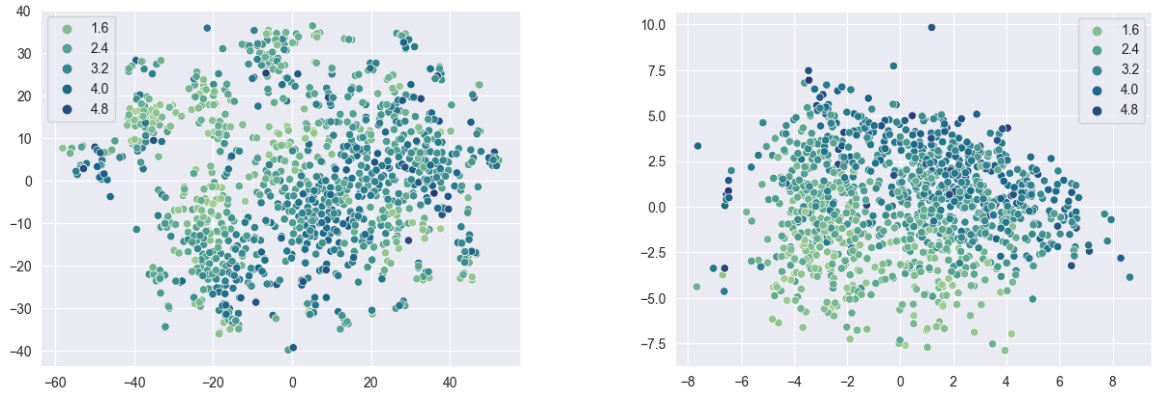
This section will report the results of the experiments described above.

4.1 Preliminary study

In the initial phase of this study, I conducted an exploration to understand the characteristics of the dataset and the superficial relationships between the set of implicit and explicit features. To begin with, I applied dimensionality reduction techniques to the dataset represented by explicit features, aiming to confirm the strong dependence of sentence complexity on linguistic characteristics. Figure 3a presents the plot of the first two components obtained from a t-SNE dimensionality reduction, while Figure 3b shows the first two components resulting from a PCA decomposition. Each point in the figures represents a sentence, with the color of the point representing the corresponding sentence complexity score.

The analysis, particularly evident in the PCA visualization, reveals a clear pattern indicating that linguistic features align the sentences along a specific direction in the feature space, which corresponds to the complexity of the sentences. To further support this finding, I trained a LASSO regressor using profiling features to predict the complexity scores, resulting in an R^2 value of 0.7227, indicating a strong relationship between the linguistic features and sentence complexity.

The second preliminary investigation focuses on the correlation between implicit and explicit features. I computed Pearson's correlation coefficient to construct a correlation matrix between the two sets of features. Figure 4 presents a heatmap representation of the correlation matrix, where non-significant correlations were set to zero. Each row in the matrix corresponds to an implicit feature,



(a) t-SNE (b) PCA

Figure 3: Visualization of the dataset represented with the set of Profiling-UD (explicit) features. The x represents the first component of the projection space, y represents the second component. Each point represents a sentence and its color is an indicator of the sentence complexity.

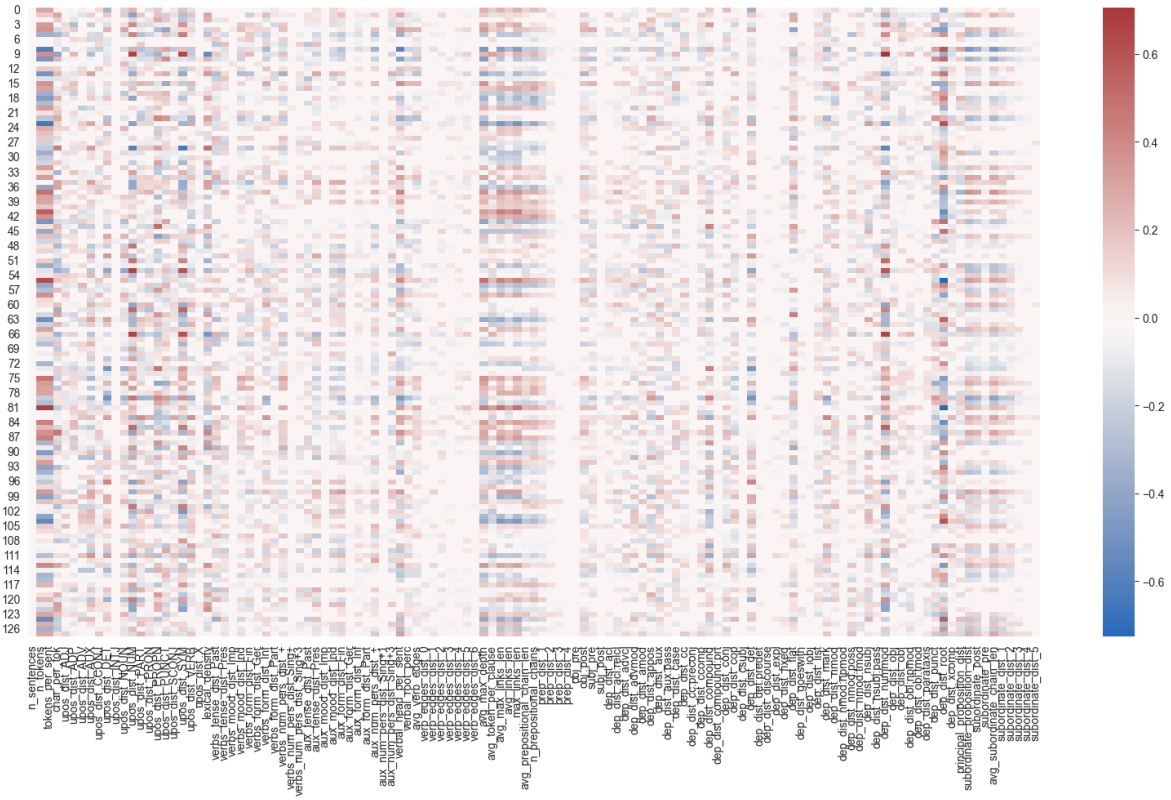
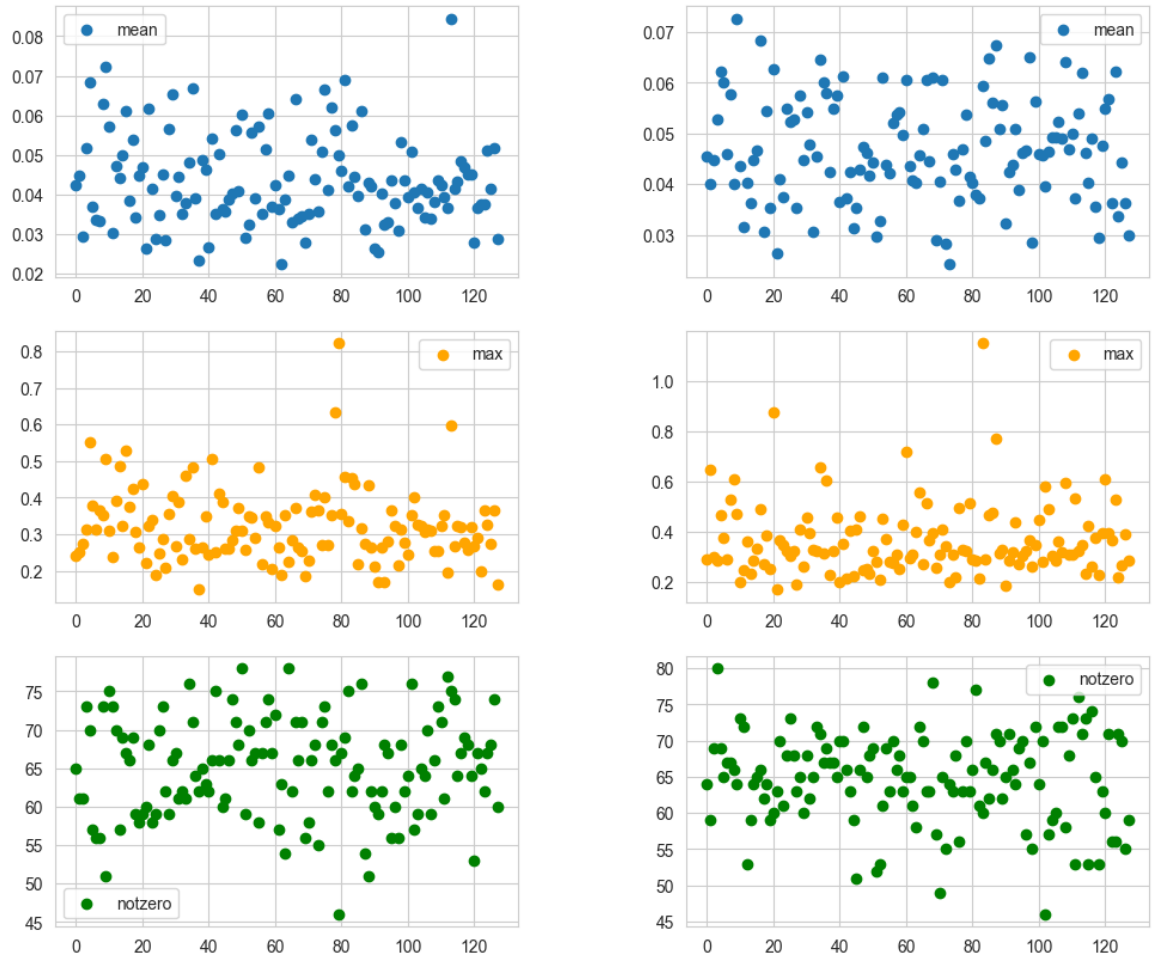


Figure 4: Correlation matrix between Profiling-UD features and implicit features extracted from the first layer of BERT-tiny.

with the row index indicating its position in the feature vector. Each column represents an explicit feature. The heatmap reveals several significant correlations, with coefficients around 0.7. In particular, the explicit features with the highest correlation coefficients include `dep_dist_root`, `dep_dist_nummod`, `n_tokens`, `tokens_per_sent`, `pos_dist_NUM`, `upos_dist_NUM`, and `upos_dist_SYM`. These features are all related to sentence length. A similar pattern emerges when extracting implicit features from the second layer of `BERT-tiny`, confirming the consistency of the findings.

4.2 Feature importance with LASSO

We will now proceed to delve into the core of the experimental investigations. As mentioned earlier, the dataset was represented using the 128 implicit features extracted from a layer of `BERT-tiny`, and these representations were utilized to train a LASSO regressor for each explicit feature. It is important to note that the focus of this experiment lies not in the regression's quality, but rather in the determination of feature importance. The LASSO regressor produces an array of coefficients that can be interpreted as a feature importance array, where features with higher coefficients (in absolute value) are considered more important. With the acquisition of 121 feature importance arrays, the next step involved consolidating them into a single array, which encapsulates the importance of each explicit feature in predicting linguistic information on a global scale. Figure 5 reports scatter plots of the global importance arrays for both sets of implicit features.



(a) Layer 1

(b) Layer 2

Figure 5: Plot of global feature importance vectors computed for implicit features extracted from Layer 1 (a) and 2 (b) of `BERT-tiny`. The x axis contains the index of the considered feature; the y axis contains the global importance score of the feature. The blue importances are computed with the *mean* method, the orange ones with the *max* method and the green ones with the *not zero* method.

For layer 1 were identified 3 outliers: feature 113 with the mean method, and features 78 and 79 with the max method. For layer 2, components 20 and 83 were identified as outliers using the max method. As no consistent pattern emerged across the three different methods, I made the decision of continuing the investigation without making any distinction between the found outliers.

4.2.1 Evaluation of the most important features

In this phase, the focus was on evaluating whether the identified outliers carried significant linguistic information. To assess this, I compared the performance of a regressor trained on complexity scores using all implicit features against using all features except for each outlier. The results are presented in the top parts of Tables 1 and 2. Surprisingly, removing the outlier features did not result in a loss of performance.

Model	$\mathbf{R^2}$
All features	0.7503
Removed random	0.7501
Removed 113	0.7501
Removed 78	0.7501
Removed 79	0.7501
Removed 113, 78, 79	0.7501
Only random	0.0152
Only 113	0.0440
Only 78	0.0523
Only 79	-0.0113
Best feature (23)	0.4512

Table 1: Layer 1

Model	$\mathbf{R^2}$
All features	0.7402
Removed random	0.7412
Removed 20	0.7412
Removed 83	0.7412
Removed 20, 83	0.7412
Only random	-0.0147
Only 20	0.0201
Only 83	0.0291
Best feature (55)	0.3338

Table 2: Layer 2

In the second experiment, I attempted to predict the complexity score using only each individual outlier feature (bottom part of Tables 1 and 2). As shown in the tables, the outlier features did not provide more information than a random feature. Interestingly, I did find a feature for each layer that, when used alone, exhibited decent predictive power. However, these features could not be identified using the LASSO regression coefficients.

The inability to identify the important features using the LASSO regression coefficients may have two underlying reasons. Firstly, it is possible that these features contain a significant amount of "non-linguistic" information, such as semantic information, which plays a crucial role in solving the sentence complexity task. Alternatively, it could be attributed to a failure of the feature selection method itself.

To address the latter problem, I attempted a more sophisticated feature selection method called ABESS (Adaptive Best Subset Selection) [9], which is a polynomial implementation of the Best Subset Selection technique. Unfortunately, even this method failed to identify the outlier features. The results of this attempt for the first layer are reported in Table 3.

Model	$\mathbf{R^2}$
All features	0.7503
Removed random	0.7501
Removed 9	0.7501
Removed 78	0.7501
Removed 79	0.7501
Removed 9, 78, 79	0.7520
Only random	0.0823
Only 9	0.2216
Only 78	0.0523
Only 79	-0.011
Best feature (23)	0.4512

Table 3: Experiment on regression when using or not using outlier implicit features found with ABESS.

4.2.2 Preliminary findings

Despite the presence of outlier features that exhibited promising individual predictive power, their contribution to linguistic information remains elusive, necessitating further investigation and alternative approaches. It is noteworthy that neither the LASSO regression coefficients nor the ABESS feature

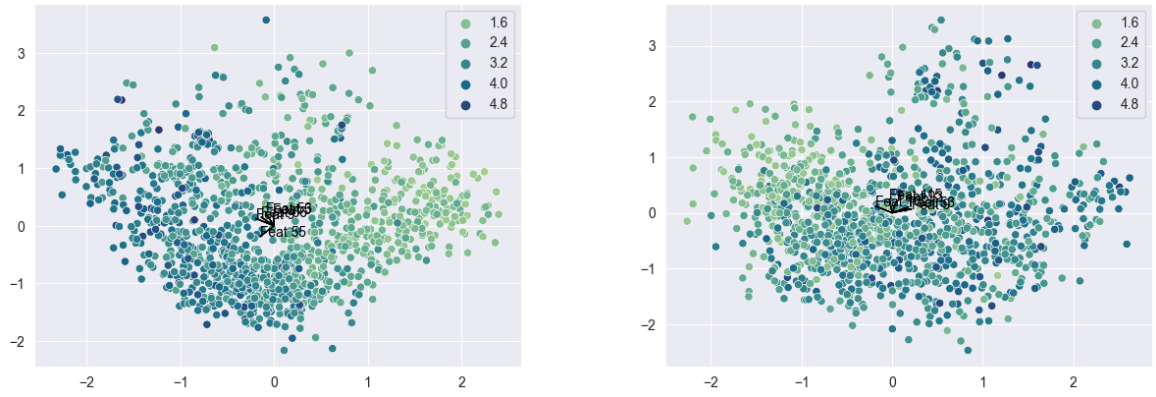
selection method successfully identified the most important features. I therefore wanted to explore potential explanations.

One striking observation is that removing a single feature or even a small set of features from the dataset description does not result in a significant loss of performance, even when the excluded feature is considered the "best" one. This suggests that individual implicit features do not carry distinct and useful information; rather, they exhibit a high degree of redundancy. This behavior indicates strong collinearity between the variables.

4.3 Overcoming collinearity: PCA

To address the collinearity issue, I employed PCA with whitening on the set of implicit features. The resulting PCA decomposition is depicted in Figure 6, showcasing the first two components extracted from implicit features for both layers of `BERT-tiny`. Each point on the plot represents a sentence, with the color indicating its complexity score. Notably, the distribution of points in this transformed space exhibits a *beautiful* gradient corresponding to sentence complexity. This suggests the potential for achieving high performance by leveraging a small subset of PCA-transformed implicit features.

In the plots, barely-visible black arrows denote the contributions of the top five original features. However, the arrow lengths appear short, indicating that the PCA components are composed of a nearly equal combination of numerous original features. Consequently, while attempting to identify the true "outlier" features, it becomes evident that re-conducting them back into a compact set of original features would be impossible. The extensive influence of multiple original features within each PCA component diminishes the prospect of isolating distinct outliers, even if they exist.



(a) Layer 1 (b) Layer 2

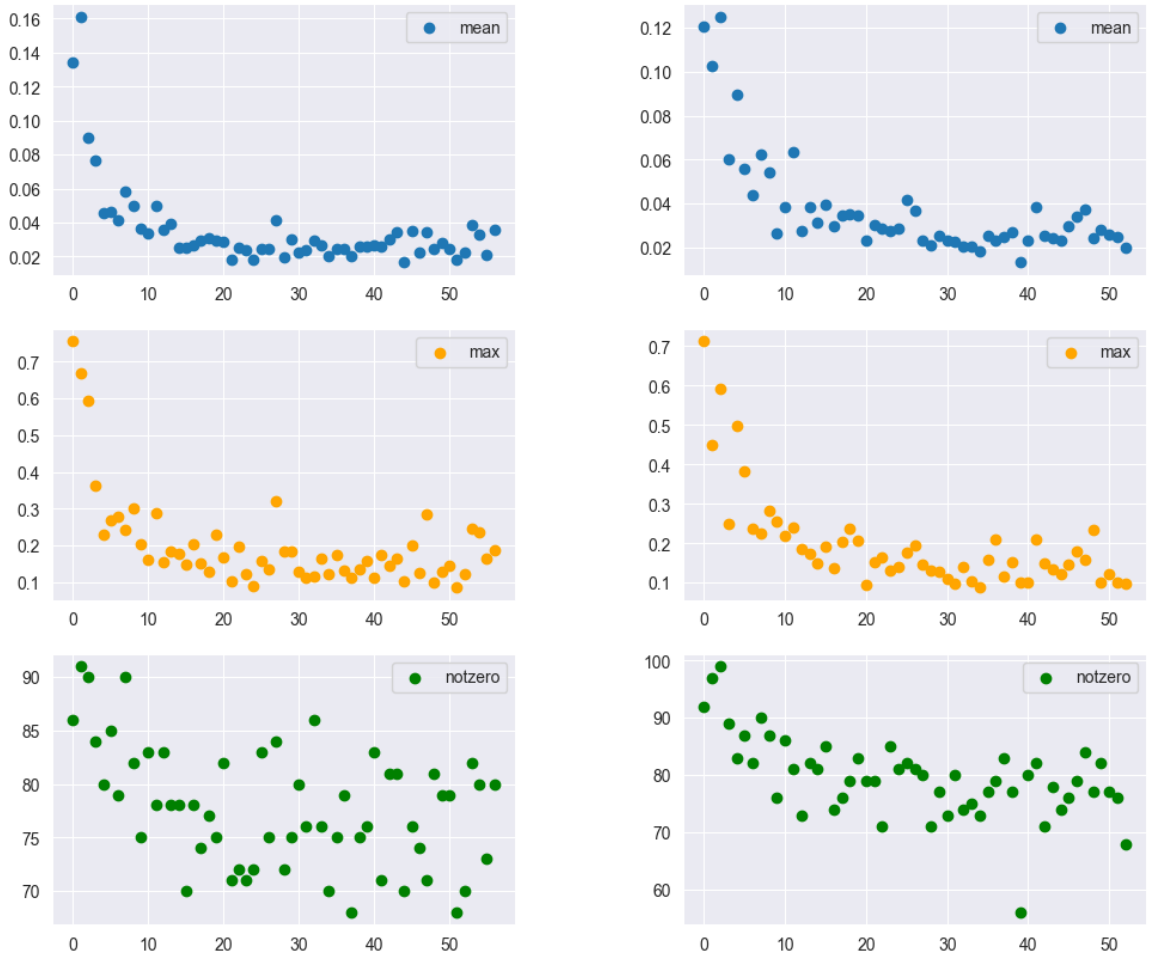
Figure 6: Plot of the first two components of PCA decomposition for the implicit features extracted from the first layer (a) and the second layer (b) of `BERT-tiny`. The black arrows represent the original features contributions.

This discovery highlights how challenging it is to unravel the influence of individual features in the PCA-transformed space. It makes it difficult to identify a small, influential set of original features. While the PCA-transformed implicit features hold promise, more investigation is needed to fully understand the relationship between these components and how they contribute to predicting sentence complexity.

4.3.1 Results with PCA

Nevertheless, I reproduced the preceding experiments using PCA-transformed features, to look for something interesting. Figure 7 reports the aggregated feature importance scores, obtained through LASSO regression on the PCA-transformed implicit features. Notably, a discernible pattern emerges, showcasing that the initial few features hold significantly higher importance compared to the subsequent ones. For the features extracted from the first layer, components 0, 1 and 2 are identified as outliers. Similarly, for the features extracted from the second layer, outliers are observed in compo-

nents 0 and 2. This trend further highlights the distinctive contribution of specific PCA components in capturing and predicting linguistic information within the transformed feature space.



(a) Layer 1 (b) Layer 2

Figure 7: Plot of global feature importance vectors computed for PCA components of implicit features extracted from Layer 1 (a) and 2 (b) of `BERT-tiny`.

Once identified the outliers I proceeded again to compute the performances on the complexity regression task, including or excluding them. Results of this experiment are reported on Table 4.

Model	$\mathbf{R^2}$
All features	0.7555
Removed random	0.7547
Removed 0	0.1424
Removed 1	0.7001
Removed 2	0.7365
Only random	-0.0153
Only 0	0.5203
Only 1	0.0285
Only 2	0.0004
Best feature (0)	0.5203

Model	$\mathbf{R^2}$
All features	0.7335
Removed random	0.7334
Removed 0	0.5284
Removed 1	0.7193
Only random	-0.0150
Only 0	0.1694
Only 2	-0.0025
Best feature (4)	0.2367

Table 4: Experiment on regression when using or not using outlier implicit features transformed with PCA for Layer 1 (left) and Layer 2 (right).

This time, at least for experiment conducted on the first layer, some of the identified outliers do carry most of the information. Unfortunately, as stated before, these outliers can't be linked to a small set of original features.

4.4 Bigger models

This section reports the summarized results of this experiments conducted on two bigger models: `BERT-mini` (consisting of 4 layers with a hidden dimension of 256) and `BERT-medium` (comprising 8 layers with a hidden dimension of 512). Table 5 and Table 6 report scores of linear regressor trained on implicit features extracted from, respectively, `BERT-mini` and `BERT-medium` varying the presence of the outliers found using the same method as before. The top block of the two tables reports the scores of the experiments conducted on the original features. This includes the scores obtained using all features and the best single feature alone. The number inside the brackets denotes the index of the best score, and if the cell content is bold, it indicates that the best feature was among the identified outliers. Scores of regressors trained removing each outlier are not reported because did not lead to decreased performances. The second block of the tables reports the experiment conducted on the PCA-transformed features. The first row simply indicated how many PCA-features are needed to explain 95% of variance, the second one reports the R^2 of a linear regressor trained on all the PCA-features, the third line contains the score obtained removing a random feature and the following lines of the block reports the scores obtained removing an outlier. The bottom block contains the results of experiments where the linear regressor was trained using only a random feature or one of the outliers. Again, the number inside the brackets contains the index of the corresponding feature (where 0 represents the first PCA component, 1 represents the second, and so on). Bold cells indicate that the considered outlier feature contributes to an R^2 improvement of more than 0.1.

	Layer 1	Layer 2	Layer 3	Layer 4
Orig. features				
All features	0.7195	0.7310	0.7321	0.7355
Best score	0.5328 (172)	0.5118 (190)	0.4429 (190)	0.1841 (139)
PCA 95% dim	114	114	111	103
All features	0.7607	0.7698	0.7715	0.7538
Removed random	0.7593	0.7705	0.7718	0.7545
Removed O1	**0.2567 (0)**	**0.3287 (0)**	**0.5845 (0)**	**0.6148 (0)**
Removed O2	**0.3649 (1)**	**0.3455 (1)**	**0.5097 (1)**	**0.6939 (1)**
Removed O3	0.7590 (2)	0.7704 (2)	**0.5276 (2)**	0.7106 (2)
Removed O4	--	--	--	0.6086 (4)
Only random	-0.0155	-0.0157	-0.0152	-0.0185
Only O1	**0.3846 (0)**	**0.3348 (0)**	**0.1387 (0)**	0.1000 (0)
Only O2	**0.2854 (1)**	**0.3061 (1)**	**0.1859 (1)**	0.0301 (1)
Only O3	-0.0155 (2)	-0.0158 (2)	**0.1692 (2)**	0.0133 (2)
Only O4	--	--	--	0.0921 (4)

Table 5: R^2 of the experiment conducted extracting implicit representations from the 4 layers of `BERT-mini`, both for the original set of features and PCA-transformed features

These findings reveal that the observed patterns and behaviors persist with the larger BERT models, aligning with the observations made for `BERT-tiny`. However, an exception arises in the case of the last three layers of `BERT-medium`, where the LASSO feature selection method successfully identified the "best feature".

Figure 8 showcases the plots of the global feature importance vectors computed using the *max* method for the 6th (left), 7th (center), and 8th (right) layers of BERT-medium. Notably, in these layers, the "best feature" (component 427) was correctly identified as an outlier. While in Layer 6, this component exhibited a substantial deviation from the others, in Layers 7 and 8, its behavior was less prominent. This suggests that the successful identification of the "best feature" in these layers may have been attributed to chance rather than a consistent pattern.

What comes clear from these results is that the smaller the considered model (and so the feature space), the more first layer's first PCA component carries linguistic information. When considering

	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7	Layer 8
Orig. features								
All features	0.5305	0.5435	0.5565	0.5540	0.5655	0.5959	0.5735	0.5263
Best score	0.4450 (400)	0.3740 (270)	0.3969 (270)	0.4108 (270)	0.6030 (427)	**0.5994 (427)**	**0.4952 (427)**	**0.3972 (427)**
PCA 95% dim	228	221	220	222	226	214	214	195
All features	0.7212	0.7281	0.7422	0.7398	0.7418	0.7417	0.7315	0.7379
Removed random	0.7211	0.7142	0.7411	0.7398	0.7367	0.7420	0.7317	0.7387
Removed O1	**0.2632 (0)**	**0.4323 (0)**	**0.3985 (0)**	**0.3955 (0)**	**0.4544 (0)**	**0.4389 (0)**	**0.5694 (0)**	**0.5977 (0)**
Removed O2	**0.0070 (1)**	**-0.0647 (1)**	**0.0664 (1)**	**0.0843 (1)**	**0.1567 (1)**	**0.3374 (1)**	**0.4817 (1)**	**0.6650 (1)**
Removed O3	0.7221 (2)	0.7300 (2)	**0.6983 (2)**	**0.6572 (2)**	**0.5570 (2)**	**0.4224 (2)**	**0.4729 (2)**	**0.6923 (2)**
Removed O4	--	--	--	--	--	--	**0.4673 (3)**	**0.6364 (4)**
Removed O5	--	--	--	--	--	--	--	**0.4544 (6)**
Only random	-0.0172	-0.0091	-0.0154	-0.0159	-0.0146	-0.0161	-0.0140	-0.0168
Only O1	**0.2607 (0)**	**0.1848 (0)**	**0.2049 (0)**	**0.2003 (0)**	**0.1768 (0)**	**0.1797 (0)**	**0.1014 (0)**	0.0868 (0)
Only O2	**0.3568 (1)**	**0.4165 (1)**	**0.3507 (1)**	**0.3368 (1)**	**0.2959 (1)**	**0.1806 (1)**	**0.1244 (1)**	0.0229 (1)
Only O3	-0.0171 (2)	-0.0170 (2)	0.0127 (2)	0.0384 (2)	0.0823 (2)	**0.1664 (2)**	**0.1371 (2)**	0.0093 (2)
Only O4	--	--	--	--	--	--	**0.1514 (3)**	0.0562 (4)
Only O5	--	--	--	--	--	--	--	**0.1775 (6)**

Table 6: R^2 of the experiment conducted extracting implicit representations from the 8 layers of `BERT-medium`, both for the original set of features and PCA-transformed features

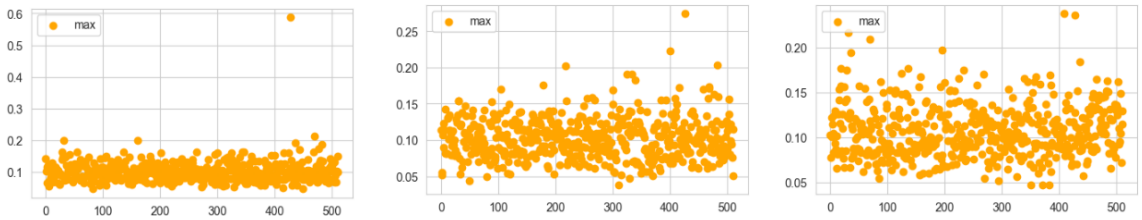


Figure 8: Scatter plot of the global feature importance vector computed with the *max* method for the 6th (left), 7th (center) and 8th (right) layer of `BERT-medium`.

bigger models, this kind of information is more spread among the components, especially in the deeper layers. This suggests that the representation of linguistic information undergoes a transformation and diffusion throughout the layers of the model, with later layers playing a more significant role in capturing complex linguistic patterns.

5 Conclusion

In this study, I conducted a series of experiments to investigate the relationship between implicit and explicit features in the context of sentence complexity prediction. To address this challenge, I explored the concept of outlier features. I trained LASSO regressors on implicit features extracted from different layers of the BERT models, identifying outliers using feature importance scores. Surprisingly, removing these outliers did not lead to a significant loss in performance, suggesting a redundancy among implicit features. Moreover, individual outlier features did not demonstrate a higher predictive power compared to random features. This raised questions about the true nature and role of these outlier features in sentence complexity prediction. To overcome the issue of collinearity among features, I applied Principal Component Analysis with whitening to the implicit feature matrices. The PCA decomposition provided a new perspective, revealing a gradient in sentence complexity along the transformed feature space. However, the PCA components were composed of a large set of original features, making impossible a mapping to a small set of original features that carry a lot of linguistic information. The feature importance scores obtained from LASSO regression on PCA-transformed features showed a diminishing trend, with the first few components carrying the most information, specially in first layers. We can conclude that probably, NLMs extract linguistic information in the first layers, in a distributed way among its implicit features set. This kind of information is then lost on deeper layers, being mixed with other types of features to capture more complex types of information.

5.1 Future Work

Considering the strong correlation between certain implicit features and explicit features related to sentence length, which itself holds considerable predictive power for sentence complexity (0.61 R^2), it would be interesting to conduct further experiments by mitigating the influence of sentence length. There are several approaches that can be explored to achieve this goal. One approach involves generating a dataset where all sentences have the same length, thereby removing the variability introduced by sentence length. By computing feature importance on this modified dataset, we can assess the significance of implicit features independent of sentence length. Another approach entails excluding explicit features related to sentence length when calculating the importance of implicit features. This method allows us to evaluate the contribution of implicit features beyond their association with sentence length, providing a clearer understanding of their unique impact on sentence complexity. Lastly, we can remove implicit features that exhibit a high correlation with sentence length from the set of implicit features. By eliminating these features, we can examine the remaining implicit features' contributions to sentence complexity, disentangling their influence from the confounding effect of sentence length.

By exploring these alternative approaches, we can gain deeper insights into the specific role of implicit features in sentence complexity prediction, separate from the effects of sentence length. This would enhance our understanding of the broader linguistic factors at play and potentially improve the accuracy and interpretability of sentence complexity models.

References

[1] D. Brunato, A. Cimino, F. Dell'Orletta, G. Venturi, and S. Montemagni. Profiling-UD: a tool for linguistic profiling of texts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7145–7151, Marseille, France, May 2020. European Language Resources Association.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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, June 2019. Association for Computational Linguistics.

[3] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[4] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[5] P. Rajpurkar, R. Jia, and P. Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

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

[8] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[9] J. Zhu, C. Wen, J. Zhu, H. Zhang, and X. Wang. A polynomial algorithm for best-subset selection problem. *Proceedings of the National Academy of Sciences*, 117(52):33117–33123, 2020.