Tracking linguistic information in transformer-based sentence embeddings through targeted sparsification

Vivi Nastase[1] and **Paola Merlo**[1,2]
[1]Idiap Research Institute, Martigny, Switzerland
[2]University of Geneva, Swizerland
vivi.a.nastase@gmail.com, Paola.Merlo@unige.ch

Abstract

Analyses of transformer-based models have shown that they encode a variety of linguistic information from their textual input. While these analyses have shed a light on the relation between linguistic information on one side, and internal architecture and parameters on the other, a question remains unanswered: how is this linguistic information reflected in sentence embeddings? Using datasets consisting of sentences with known structure, we test to what degree information about chunks (in particular noun, verb or prepositional phrases), such as grammatical number, or semantic role, can be localized in sentence embeddings. Our results show that such information is not distributed over the entire sentence embedding, but rather it is encoded in specific regions. Understanding how the information from an input text is compressed into sentence embeddings helps understand current transformer models and help build future explainable neural models.

1 Introduction

In the quest for understanding transformer-based models, much work has been dedicated to uncover what kind of information is encoded in the model's various layers and parameters. These analyses have provided several enlightening insights: (i) different types of linguistic information – e.g. parts of speech, syntactic structure, named entities – are selectively more evident at different layers of the model (Tenney et al., 2019a; Rogers et al., 2020), (ii) subnetworks can be identified that seem to encode particular linguistic functionalities (Csordás et al., 2021), and (iii) fine-tuning for specific tasks can be focused on very small subsets of parameters, on different parts of a model's layers (Panigrahi et al., 2023). While these analyses and probes have focused on the insides of the models, mostly their parameters and layers, testing their impact is usually done by using the output of the model, namely token or sentence embeddings, to solve specific tasks. The link between the inside of the model and its outputs is usually not explicitly investigated.

We ask several facets of this question here: how are the internally-detected information types and structures reflected in the model's output? And how are arbitrarily long and complex sentences encoded systematically in a fixed-sized vector?

Understanding what kind of information the sentence embeddings encode, and how, has multiple benefits: (i) it connects internal changes in the model parameters and structure with changes in its outputs; (ii) it contributes to verifying the robustness of models and whether or not they rely on shallow or accidental regularities in the data; (iii) it narrows down the field of search when a language model produces wrong outputs, and (iv) it helps maximize the use of training data for developing more robust models from smaller textual resources.

Transformer-based models usually use a token-focused learning objective, and have a weaker supervision signal at the sentence level – e.g. a next sentence prediction (Devlin et al., 2018), or sentence order information (Lan et al., 2019). Despite this focus, high performance in a variety of tasks (using raw or fine-tuned sentence embeddings) as well as direct probing shows that sentence representations encode a variety of linguistic information (Conneau et al., 2018). On the other hand, direct exploration of BERT sentence embeddings has also shown that they contain mostly shallow information, related to sentence length and lexical variation, and that many of their dimensions are correlated, indicating that either information is redundantly encoded, or that not all dimensions encode useful information (Nikolaev and Padó, 2023). Some of this preexisting work assumes that sentence embeddings encode information in an overt manner, for example, each principal component dimension is responsible for encoding some type of information.

We adopt the different view that information in

Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024), pages 203–214

sentence embeddings may be encoded in merged layers, in a manner similar to audio signals being composed of overlapping signals of different frequencies. We hypothesize that each such layer may encode different types of information. We aim to test this hypothesis and check (i) whether we can separate such layers, and (ii) investigate whether information about specific chunks in a sentence –noun,verb, or prepositional phrases– is encoded in different layers and parts of a sentence embedding.

We perform our investigation in an environment with data focused on specific grammatical phenomena, while displaying lexical, structural and semantic variation, and a previously developed system that has been shown to detect the targeted phenomena well (Nastase and Merlo, 2024). The system is a variational encoder-decoder, with an encoder that compresses the information in the input into a very low-dimensional latent vector. Nastase and Merlo (2024) have shown that the sentence embeddings, and their compressed representations on the latent layer, encode information about chunks – noun, verb, prepositional phrases – and their linguistic properties.

The current study investigates the general hypothesis indicated above by specifically exploring two new research questions in this setting:

1. Whether a targeted sparsification of the system maintains a high performance on the task, indicating that information about chunks in the sentence is localizable.

2. Contingent on the answer to the first question, we trace back the signal from the latent layer to the input sentence embeddings, and analyze how specific differences in chunk properties – different number of chunks, or chunks that differ from each other only on one property (e.g. grammatical number) – are localized and reflected in the sentence embeddings.

The code and data are available at https://github.com/CLCL-Geneva/BLM-SNFDisentangling.

2 Related work

Sentence embeddings Transformer models induce contextual token embeddings by passing the embedding vectors through successive layers using multi-head attention that allows for tokens to influence each other's representation at each successive step (Vaswani et al., 2017). The model focuses on the token embeddings, as the tokens expected on the output layer provide the training signal. There are numerous variations on the BERT (Devlin et al., 2018) transformer model[1], that vary in the way the models are trained (Liu et al., 2019), how they combine (or not) the positional and token embeddings (He et al., 2020), how the input is presented to the model (Liu et al., 2019; Clark et al., 2020). With regards to the sentence-level supervision signal, BERT (Devlin et al., 2018) uses the next sentence prediction objective, ALBERT (Lan et al., 2019), aiming to improve coherence, uses sentence order prediction. It is more common to further train or fine-tune a pre-trained model to produce sentence embeddings fitting specific tasks, such as story continuation (Ippolito et al., 2020) or sentence similarity (Reimers and Gurevych, 2019).

Electra (Clark et al., 2020) does not have a sentence-level objective, but it relies on replaced token detection, which relies on the sentence context to determine whether a (number of) token(s) in the given sentence were replaced by a generator sample. This leads to sentence embeddings that perform well on tasks such as Question Answering, or detecting verb classes (Yi et al., 2022).

Probing embeddings and models for linguistic information Most work investigating the kind of knowledge captured by transformer-based models have focused on analysing the architecture of the model (Tenney et al., 2019b; Rogers et al., 2020) to determine the localization and flow of information through the model's layers. There is also much work on analyzing the induced token embeddings to determine what kind of linguistic information they encode, such as sentence structure (Hewitt and Manning, 2019), predicate argument structure (Conia et al., 2022), subjecthood and objecthood (Papadimitriou et al., 2021), among others. Testing whether sentence representation contain specific types of linguistic information has been done using task (or information)-specific classifiers (Adi et al., 2017; Conneau et al., 2018; Goldberg, 2019; Wilson et al., 2023). Opitz and Frank (2022) aim to map subsets of dimensions of fine-tuned sentence embeddings to semantic features.

Sparsification Deep learning models have billions of parameters. This makes them not only incomprehensible, but also expensive to train. The

[1] https://huggingface.co/docs/transformers/en/model_summary

lottery ticket hypothesis (Frankle and Carbin, 2018) posits that large networks can be reduced to subnetworks that encode efficiently the functionality of the entire network. Detecting functional subnetworks can be done *a posteriori*, over a pre-learned network to investigate the functionality of detected subnetworks (Csordás et al., 2021), the potential compositionality of the learned model (Lepori et al., 2023), or where task-specific skills are encoded in a fine-tuned model (Panigrahi et al., 2023).

Instead of learning a sparse network over a pre-learned model, Cao et al. (2021) use a pruning-based approach to finding subnetworks in a pre-trained model that performs some linguistic task. Pruning can be done at several levels of granularity: weights, neurons, layers. Their analyses confirm previous investigations of the types of information encoded in different layers of a transformer (Conneau et al., 2018). Conmy et al. (2023) introduce the Automatic Circuit DisCovery (ACDC) algorithm, which adapts subnetwork probing and head importance score for pruning to discover circuits that implement specific linguistic functions.

Sparsification can also be achieved using L_0 regularization, as the pruning would be done directly during training by encouraging weights to become exactly zero. Louizos et al. (2018); Savarese et al. (2020), among others, implement solutions to the issue that L_0 regularization is non-differentiable, and test it on image classification.

The cited work focuses on the parameters of the model, and sparsification approaches aiming to detect the subnetworks to which specific skills or linguistic information can be ascribed. Our focus, instead, is the output of transformer-based models, in particular sentence embeddings, which we investigate using targeted sparsification.

3 Approach overview

We investigate whether we can identify specific sentence properties in sentence embeddings. Nastase and Merlo (2024) have shown that using an encoder-decoder architecture, sentence embeddings can be compressed into a latent representation that preserves information about chunks in a sentence, and their properties necessary to solve a specific linguistic task.

We first test whether we can sparsify this architecture in a targeted manner, such that each region of the sentence embedding contributes a signal to only one unit of the latent layer. This allows us to isolate different parts of the sentence embedding.

After establishing that sparsification does not lead to a dramatic drop in performance, we trace back the signal from the latent layer to the sentence embeddings, and test whether we can localize information about how different numbers of chunks, or chunks with different properties, are encoded.

In the final step, we use the sparse encoder-decoder sentence compression system as the first in a two-layer system used to solve language tasks – called Blackbird Language Matrices (Merlo, 2023) – that require chunk and chunk properties information. The first layer will compress each sentence into a very small latent vector, and this representation is then used on the second layer to solve a pattern detection problem that relies on information about chunks in a sentence and their pattern across a sequence of sentences.

4 Data

We use two data types: (i) a dataset of sentences with known chunk structure and chunk properties, (ii) two datasets representing two multiple-choice problems, whose solution requires understanding the chunk structure and chunk properties of the sentences in each instance.

4.1 A dataset of sentences

We start with an artificially-created set of sentences built from noun, prepositional and verb phrases. Each sentence has one of the following structures: NP [PP_1 [PP_2]] VP , where the parentheses surround optional structure. Each chunk can have singular or plural form, with agreement between the first NP (the subject) and the VP. This leads to 14'336 sentences with one of 14 patterns.

The dataset consists of ordered pairs of one input sentence and N (=7) output sentences, extracted from the set described above. Only one of the output sentences has the same chunk pattern as the input sentence, and is considered as the correct output. We select 4004 instances uniformly distributed over the 14 patterns, which are split into train:dev:test – 2576:630:798.

4.2 Multiple Choice Problems: Blackbird Language Matrices

Blackbird Language Matrices (BLMs) (Merlo, 2023) are language versions of the visual Raven Progressive Matrices (RPMs). Like the RPMs, they are multiple-choice problems. The input is a sequence of 7 sentences built using specific rules, and

BLM agreement problem

CONTEXT TEMPLATE			
NP-sg	PP1-sg		VP-sg
NP-pl	PP1-sg		VP-pl
NP-sg	PP1-pl		VP-sg
NP-pl	PP1-pl		VP-pl
NP-sg	PP1-sg	PP2-sg	VP-sg
NP-pl	PP1-sg	PP2-sg	VP-pl
NP-sg	PP1-pl	PP2-sg	VP-sg

ANSWER SET	
NP-sg PP1-sg et NP2 VP-sg	Coord
NP-pl PP1-pl NP2-sg VP-pl	correct
NP-sg PP1-sg VP-sg	WNA
NP-pl PP1-pl NP2-pl VP-sg	AE_V
NP-pl PP1-sg NP2-pl VP-sg	AE_N1
NP-pl PP1-pl NP2-sg VP-sg	AE_N2
NP-pl PP1-sg PP1-sg VP-pl	WN1
NP-pl PP1-pl PP2-pl VP-pl	WN2

BLM verb alternation problem

CONTEXT TEMPLATE			
NP-Agent	Verb	NP-Loc	PP-Theme
NP-Theme	VerbPass	PP-Agent	
NP-Theme	VerbPass	PP-Loc	PP-Agent
NP-Theme	VerbPass	PP-Loc	
NP-Loc	VerbPass	PP-Agent	
NP-Loc	VerbPass	PP-Theme	PP-Agent
NP-Loc	VerbPass	PP-Theme	

ANSWER SET	
NP-Agent Verb NP-Theme PP-Loc	CORRECT
NP-Agent *VerbPass NP-Theme PP-Loc	AGENTACT
NP-Agent Verb NP-Theme *NP-Loc	ALT1
NP-Agent Verb *PP-Theme PP-Loc	ALT2
NP-Agent Verb *[NP-Theme PP-Loc]	NOEMB
NP-Agent Verb NP-Theme *PP-Loc	LEXPREP
*NP-Theme Verb NP-Agent PP-Loc	SSM1
*NP-Loc Verb NP-Agent PP-Theme	SSM2
*NP-Theme Verb NP-Loc PP-Agent	AASSM

Figure 1: Structure of two BLM problems, in terms of chunks in sentences and sequence structure.

the correct answer fits within the sequence defined by these rules. The incorrect options are built by corrupting some of the underlying generating rules of the input sentence sequence. Solving the problem requires identifying the entities (the chunks), their relevant attributes (their morphological or semantic properties) and their connecting operators.

We use two BLM datasets: (i) BLM-AgrF – subject verb agreement in French (An et al., 2023), and (ii) BLM-s/lE – the spray-load verb alternations in English[2] (Samo et al., 2023). The structure of these datasets – in terms of the sentence chunks and sequence structure – is shown in Figure 1.

Datasets statistics Table 1 shows the datasets statistics. Each set is split 90:10 into train:test subsets, and then we randomly sample 2000 instances as train data. 20% of the train data is used for development. Types I, II, III correspond to different amounts of lexical variation within an instance.

	Subj.-verb agr.	Verb alternations	
		ALT-ATL	ATL-ALT
Type I	2000:252	2000:375	2000:375
Type II	2000:4866	2000:1500	2000:1500
Type III	2000:4869	2000:1500	2000:1500

Table 1: Train:Test statistics for the two BLM problems.

To solve a BLM instance, the system processes the input sentence sequence and outputs a sentence representation that will be compared to the representation of the sentences in the answer set. The candidate answer closest to the generated sentence representation will be considered the correct one.

We run the experiments on the BLMs for agreement and on the verb alternation BLMs. While the information necessary to solve the agreement task is more structural, solving the verb alternation task requires not only structural information on chunks, but also semantic information, as syntactically similar chunks play different roles in a sentence.

5 Experiments

We present a progression of experiments.

1. Using the dataset of sentences with known chunk structure, we test whether a sparse variational encoder-decoder system can distill information about the chunk structure of a sentence from its embedding.

2. We analyze the sparse model, and trace the information from the latent layer back to the sentence embedding to understand where in the sentence embeddings these differences are encoded.

3. We combine the sparsified variational encoder-decoder with another VAE-like layer to solve the BLM tasks, and test whether the latent layer sentence encodings maintain information useful for the tasks.

All experiments use Electra (Clark et al., 2019)[3]. We use as sentence representations the embedding of the [CLS] token, reshaped as a two dimensional array with shape 32x24.

[2]Agent-Location-Theme (ALT) – Agent-Theme-Location (ATL)

[3]Electra pretrained model: google/electra-base-discriminator

The experiments are analyzed through the output of the system, in terms of average F1 score over three runs. For the investigations of the sentence embeddings, we also analyze the compressed vectors on the latent layer, to determine whether chunk patterns are encoded in these vectors. If these vectors cluster by the chunk pattern of the corresponding sentences it will indicate that sentence chunk patterns were indeed detected and are encoded differently in the latent layer.

5.1 Sparsification

Nastase and Merlo (2024) have shown that sentence embeddings contain information about the chunk structure and their properties using an encoder-decoder architecture that compresses the relevant information into a small latent layer. They build on Nastase and Merlo (2023) who show that reshaping a sentence embedding from the commonly used one-dimensional array to a two-dimensional representation allows grammatical information to become more readily accessible.

We adopt the system of (Nastase and Merlo, 2024), with the same architecture (including number of CNN channels and kernel size), and sparsify it, to determine whether specific information can be localized in sentence embeddings. The encoder of the system consists of a CNN layer followed by a FFNN, that compresses the information into a latent layer, as illustrated in Figure 2.

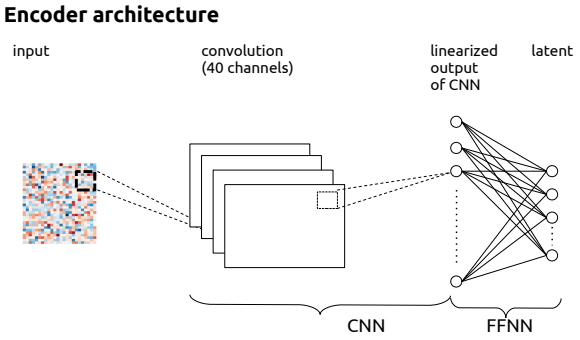


Figure 2: Details of the encoder architecture

The CNN layer in the encoder detects a different pattern in the sentence representation on each of its 40 channels. The linear layer compresses the linearized output of the CNN into a very small latent layer (length 5). A vector is sampled from this, and then decoded into a sentence representation using a decoder which is a mirror of the encoder.

An instance consists of an input sentence s, and 7 output sentences, only one of which has the same chunk structure as the input and is considered the correct one (section 4.1). The aim is to guide the system to capture information about the chunk structure of the sentences in the latent layer, by using a max-margin loss function that assigns a higher score to the correct option relative to the others. Formally, if e_s is the embedding of the input sentence s, $\hat{e}_s$ is the embedding output by the decoder, e_c is the embedding of the correct option and e_i, $i = 1, 6$ are the embeddings of the other options, and mm is the maxmargin function, then:

$$loss(s) = mm(\hat{e}_s, e_c, \{e_i | i = 1, 6\}) + KL(z_s || \mathcal{N}(0, 1))$$

$$mm(\hat{e}_s, e_c, e_i) = max(0, 1 - score(\hat{e}_s, e_c) + \sum_{i=1}^{6} score(\hat{e}_s, e_i)/6)$$

We want to sparsify this network in a targeted way: we enforce that each output unit from the CNN layer will contribute to only one unit in the latent layer. Figure 3 illustrates the process.

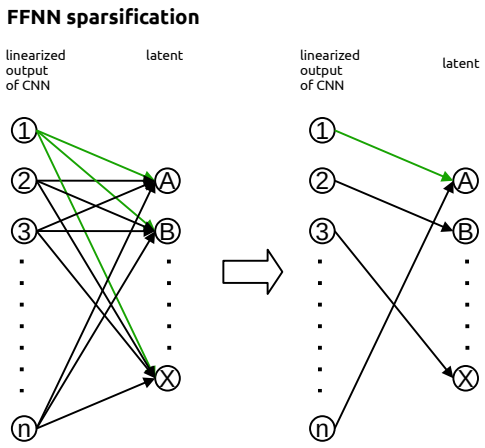


Figure 3: Separating linguistic signals by masking the one-layer FFNN

To enforce this behaviour, we use an approach inspired from sparsification (Savarese et al., 2020) and subnetworking (Lepori et al., 2023). Instead of considering the output of the CNN as the input layer of a linear network, we make each CNN output unit the input of a separate linear network, connected to the latent layer. We apply a mask m to the weights W of this network, and compute a masked weight matrix $W_m = W \times softmax(M/\tau)$, where τ is a temperature parameter used to push the softmax function towards a one-hot vector.

We use a kernel 15x15[4] and equal stride (15x15) to have a very clear separation of the information flow from the sentence embedding to the latent

[4]We adopt the size of the kernel from previous work.

layer. This will ensure our sparsification desideratum, and the learned network will have a particular configuration: if N_{CNN} is the set of output nodes from the CNN, and N_L are the nodes on the latent layer, then the sets of CNN output nodes connected to each of the latent units are disjunct:

$$\forall n_l \in N_L, S^l_{CNN} = \{n_c \in N_{CNN} | W_m(n_l, n_c) > 0\}$$

$$\text{and if } i \neq j \text{ then } S^i_{CNN} \cap S^j_{CNN} = \varnothing$$

Sparsification results Despite the fact that this type of sparsification is very harsh, and channels the information from the sentence embedding into very few paths on the way to the latent layer, the results in terms of average F1-score/standard deviation over three runs without 0.997 (0.0035) and with sparsification 0.977 (0.0095) are close. While this difference is rather small, we notice a bigger difference in the latent layer. Figure 5 shows the TSNE projections of the latent layers. As can be seen, while the full network shows a very clear and crisp separation of latents that encode different chunk patterns – with a 0.9928/0.0101 F1 macro-average/standard deviation – when sparsifying the information is slightly less crisp in the 2D TSNE projection, but still high F1 macro-average/standard deviation (0.9886/0.0038)

5.2 Localizing linguistic information in sentence embeddings

We approach the isolation of linguistic information with the following intuition: on each channel, the CNN discovers different patterns in various regions of the sentences. Some combination of these patterns – i.e. some combinations of signals from the CNN output – encode specific properties of the sentences. These signals eventually reach the latent layer. Previous experiments have shown that this latent layer contains information about chunks and their properties. Working backwards from the latent layer to the sentence embedding – through the CNN output layer, the different channels and sentence embedding regions – helps us trace back where the biggest changes are when the input sentences have different properties.

To verify whether specific linguistic information, like different number of chunks, or different chunk properties, is encoded in different regions of the sentence embeddings, we analyse the distribution of values in each network node in the encoder, namely the CNN output nodes N_{CNN} and the latent nodes N_L.

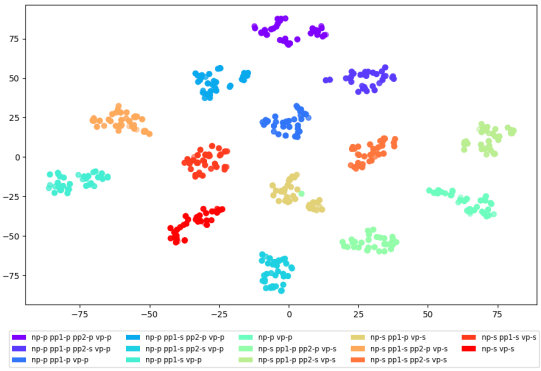


Figure 4: TSNE projection of the latent layer for encoder-decoder with full network connections.

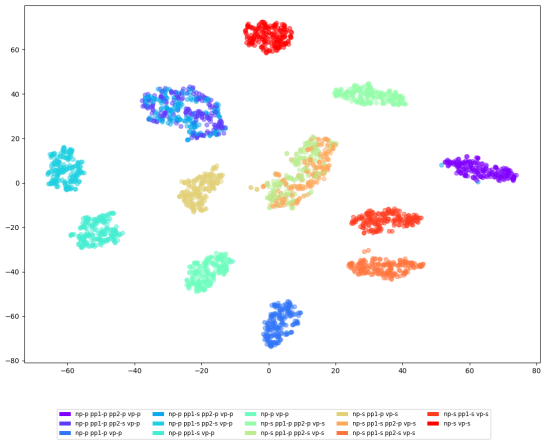


Figure 5: TSNE projection of the latent layer for sparsified encoder-decoder.

We denote S_p the set of input sentences that share the same chunk pattern p (for instance, p = "NP-s VP-s"). We pass their sentence embeddings through the learned encoder, and gather the values in each CNN output node:

$$V^p_{CNN} = \{V^p_{CNN}(n_c) | n_c \in N_{CNN}\}$$
$$V^p_{CNN}(n_c) = \{val_{n_c}(s) | s \in S_p\}$$

and $val_{n_c}(s)$ is the value in the CNN output node n_c when the input is the embedding of sentence s.

To check for differences in how sentence with different patterns are encoded, we will look at sets of sentences S_{p_1} and S_{p_2} where p_1 and p_2 are patterns that differ minimally. We consider three such minimal differences:

length one pattern has an extra (or one less) chunk than the other but are otherwise identical (*np-s vp-s* vs. *np-s pp1-s vp-s*),

grammatical number the two patterns have the same number of chunks, but one (and only one) chunk has a different grammatical number than the other (*np-s pp1-s vp-s* vs. *np-s pp1-p vp-s*),

subject-verb number alternation the two patterns are identical except in the grammatical number of the subject and verb (*np-s pp1-s vp-s* vs. *np-p pp1-s vp-p*).

To compare how chunk information is encoded in sentences that have different patterns p_1 and p_2, we compare the sets of values in each CNN output node n_c: $V^{p_1}_{CNN}(n_c)$ and $V^{p_2}_{CNN}(n_c)$. If these value distributions are very different, this is an indication that the area of a sentence embedding where the signal to n_c is coming from is involved in encoding the type of information that is different between p_1 and p_2.

We perform this analysis in two steps: (i) a filtering step that eliminates from the analysis the CNN output nodes that do not encode differences in behaviour between patterns, and (ii) a quantification of the differences in the values in the node for different patterns.

The filtering step is performed using a two-sample Kolmogorov-Smirnov test (Hodges, 1958),[5] which provides information whether two samples come from the same distribution. As we are interested in the CNN output nodes where the value distributions are different when the inputs are sentences with different patterns, we will filter out from the analysis the nodes n_c where the sets of values $V^{p}_{CNN}(n_c)$ come from the same distribution for all patterns p represented in the data.

For the remaining CNN output nodes, we project the value distributions onto the same set of bins, and then quantify the difference using cosine distance. Specifically, we determine the range of values for V^{p}_{CNN} for all patterns p – $min_{V_{CNN}}, max_{V_{CNN}}$, and split it into 100 bins. For each CNN output node n_c and pattern p we make a value distribution vector $v^{p}_{n_c}$ from the node's set of values $V^{p}_{CNN}(n_c)$, w.r.t. the 100 bins.

We then compute a score for every pair of minimally different patterns p_1, p_2 for each node n_c as the cosine distance:

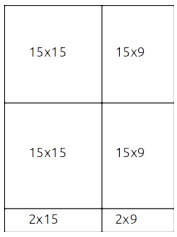
$score_{n_c}(p_1, p_2) = 1 - cos(v^{p_1}_{n_c}, v^{p_2}_{n_c})$

This score quantifies how different a region of the sentence embedding is when encoding sentences with different chunk patterns.

Localization results A first clue that information related to chunk patterns in a sentence is localized is the fact that the filtering step using the two-sample Kolmogorov-Smirnov test leads to the removal of 83 CNN output nodes out of the 240 (34%).

For the remaining nodes where differences in value distributions between different sentence patterns exist, we compute the cosine distance between pairs of minimally different patterns with respect to grammatical number, length and subject-verb number alternations. Figure 6 shows the differences in value distributions in each CNN output nodes from each channel – channels are reprezented on the y-axis, and the 5 latent units on the x-axis in different colours. A stronger colour indicates a stronger effect. More detailed plots are included in Figure 9 in the appendix.

These plots indicate that there are few channel-sentence region combinations that encode differences in chunk structure in the input sentences. While in the figure the sentence areas are illustrated with equal sizes, the regions are presented transposed for space considerations, and they have the shapes shown in the adjacent figure. The chunks and the chunk information seems to be encoded in the bottom part of the sentence embedding, and much of it in the bottom 2x24 area.



5.3 BLM tasks

To further test whether task specific information is robust to sparsification, we use the two-level variational encoder-decoder depicted in Figure 8.

An instance for a BLM task consists of a tuple, comprising a sequence of sentences $S = \{s_i | i = 1, 7\}$ as input, and an answer set with one correct answer a_c, and several incorrect answers a_{err}. The sentence level of the 2-level encoder-decoder compresses the sentence embeddings of each of the sentences in the input sequence into a small latent vector. The sampled latent representations are then used as the representations of the sentences in the input sequence. This sequence representation is passed as input to the BLM-level encoder-decoder, it is compressed into a new latent layer, and the sampled vector is then decoded into a sentence representation that best matches the representation of the correct answer.

BLM task results We evaluate the performance of the sparsified 2-level VAE on the BLM tasks. Only the first level of the VAE, the one processing individual sentences, is sparsified as described

[5] We use the ks_2samp test in the scipy Python package

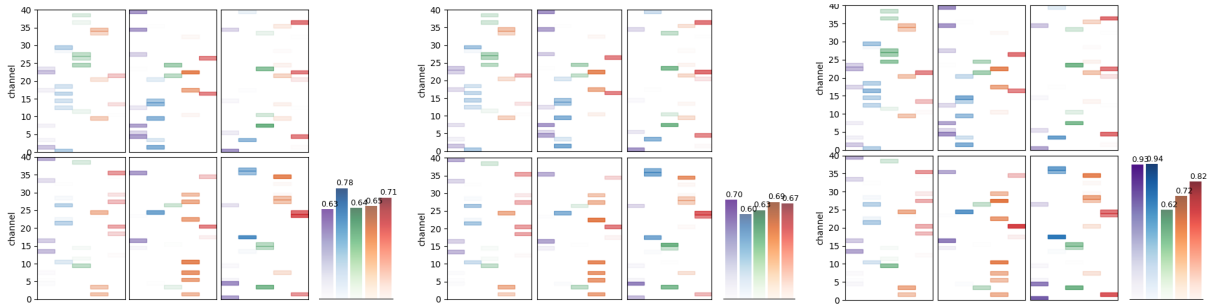


Figure 6: Average cosine distance between value distributions in each CNN output node (i.e. each node corresponding to the application of the kernel from each channel on the sentence embeddings, according to the kernel size and stride) for sets of sentences with minimally different patters: (left) patterns differ in only one grammatical number attribute for one chunk, (middle) patterns differ only in length, (right) patterns differ only in the number of the subject and verb. Each panel corresponds to one region of the sentence embedding the size of the kernel. The y-axis represents the channels of the CNN. The x-axis represents the latent units in different colours (the stronger the color, the higher the value, max = 1), and the pairs of compared patterns represented as adjacent rectangles.

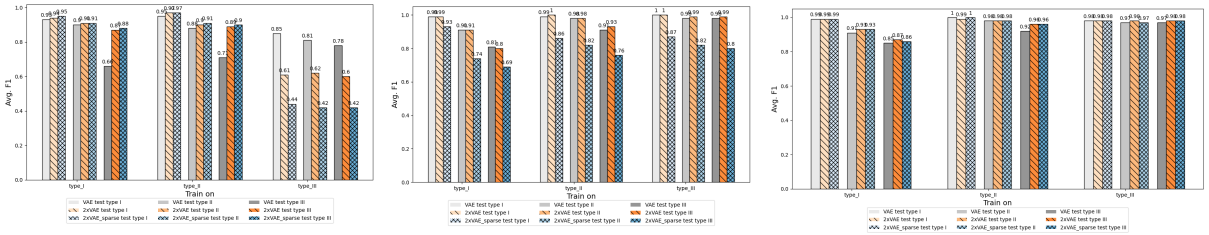


Figure 7: Results in term of average F1 scores over 3 runs, for the BLM agreement (1) and verb alternations ALT-ATL (2) and ATL-ALT (3)

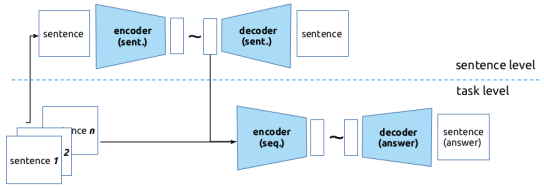


Figure 8: A two-level variational encoder-decoder: the top level compresses the sentence embeddings into a latent layer, and the bottom level uses the compressed sentence representations to solve the BLM tasks.

in section 5.1. Figure 7 shows the performance of three system variations: (i) a one-level VAE that processes the input sequence of sentences and produces a sentence representation, (ii) the two-level VAE described in more detail in (Nastase and Merlo, 2024), (iii) the sparsified version of the sentence level VAE in the two-level VAE. As in the previous experiments, sparsification does not cause harsh drops in performance for either of the two BLM tasks. The reason for this is the same reason we chose this particular data for experiments: solving the task relies on the system having information about the chunks in the sentence, and their properties. As long as that type of information is preserved, the tasks can be solved successfully.

We note two main changes however. In the agreement task, the sparsified system registers a drop in performance when trained on maximally lexically different data (type III). The two-level system without sparsification also registers such a drop in comparison with the baseline one-level encoder decoder. Both these effects may be due to the ambiguous supervision signal at the sentence level of the system: while using type I and type II data with little lexical variation, it is easier for the system to focus on structural differences between the correct and incorrect output options. When using type III data with much lexical variation, it is not clear for the system what is the relevant dimension of difference between the output options.

In the verb alternation task, previous results on predicting the Agent-Theme-Location or the Agent-Location-Theme alternation produced very similar results. This is not the case here, but understanding why this happens requires additional analysis.

6 Conclusions

Our aim was to understand how information is encoded in sentence embedding, given that previous

work has shown that various types of linguistic information is encoded in a model's layers and parameters. We investigated this question using a dataset of sentences with specific chunk structure, and two multiple-choice problems that require information about sentence chunks and their properties to be solved successfully. We have shown that using a sparsified encoder-decoder system, the sentence representations can be compressed into a latent layer that encodes chunk structure properties. We then traced back the signal from the latent layer to the sentence embedding, to detect which areas of the sentence embeddings change the most when comparing sentences with different chunk patterns. This analysis shows that such information is captured by a small number of channel-sentence area combinations. Further experiments with the two multiple-choice tasks have confirmed that chunk information and their grammatical properties (for the agreement BLM) and chunk information and their semantic role properties (for the verb alternation BLM) are captured by the sparsified sentence compression level. We envision further analyses to see where the differences between chunk patterns that have different semantic roles are encoded, and get closer to decoding the sentence embeddings.

7 Limitations

We have explored sentence embeddings using an artificially constructed dataset with simple chunk structure. To check how this kind of information is localized, we started from a previously developed system that showed high performance in distinguishing the patterns of interest. We have not changed the system's parameters (such as the kernel size of the CNNs), and have not performed additional parameter search to narrow down the locations to smaller regions. We plan to address sentence complexity issues and parameters for narrower localization of information in future work.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Aixiu An, Chunyang Jiang, Maria A. Rodriguez, Vivi Nastase, and Paola Merlo. 2023. BLM-AgrF: A new French benchmark to investigate generalization of agreement in neural networks. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1363–1374, Dubrovnik, Croatia. Association for Computational Linguistics.

Steven Cao, Victor Sanh, and Alexander Rush. 2021. Low-complexity probing via finding subnetworks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 960–966, Online. 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.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Simone Conia, Edoardo Barba, Alessandro Scirè, and Roberto Navigli. 2022. Semantic role labeling meets definition modeling: Using natural language to describe predicate-argument structures. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4253–4270, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. In *Advances in Neural Information Processing Systems*, volume 36, pages 16318–16352. Curran Associates, Inc.

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.

Róbert Csordás, Sjoerd van Steenkiste, and Jürgen Schmidhuber. 2021. Are neural nets modular? inspecting functional modularity through differentiable weight masks. In *Int. Conf. on Learning Representations (ICLR)*, Virtual only.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Training pruned neural networks. *CoRR*, abs/1803.03635.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced BERT with disentangled attention. *CoRR*, abs/2006.03654.

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.

Joseph L. Hodges. 1958. The significance probability of the smirnov two-sample test. *Arkiv für Matematik*, 3:469–486.

Daphne Ippolito, David Grangier, Douglas Eck, and Chris Callison-Burch. 2020. Toward better storylines with sentence-level language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7472–7478, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Michael Lepori, Thomas Serre, and Ellie Pavlick. 2023. Break it down: Evidence for structural compositionality in neural networks. *Advances in Neural Information Processing Systems*, 36:42623–42660.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning sparse neural networks through l_0 regularization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Paola Merlo. 2023. Blackbird language matrices (BLM), a new task for rule-like generalization in neural networks: Motivations and formal specifications. *ArXiv*, cs.CL 2306.11444.

Vivi Nastase and Paola Merlo. 2023. Grammatical information in BERT sentence embeddings as two-dimensional arrays. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 22–39, Toronto, Canada. Association for Computational Linguistics.

Vivi Nastase and Paola Merlo. 2024. Are there identifiable structural parts in the sentence embedding whole?

Dmitry Nikolaev and Sebastian Padó. 2023. The universe of utterances according to BERT. In *Proceedings of the 15th International Conference on Computational Semantics*, pages 99–105, Nancy, France. Association for Computational Linguistics.

Juri Opitz and Anette Frank. 2022. SBERT studies meaning representations: Decomposing sentence embeddings into explainable semantic features. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 625–638, Online only. Association for Computational Linguistics.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. 2023. Task-specific skill localization in fine-tuned language models. In *International Conference on Machine Learning*, pages 27011–27033. PMLR.

Isabel Papadimitriou, Ethan A. Chi, Richard Futrell, and Kyle Mahowald. 2021. Deep subjecthood: Higher-order grammatical features in multilingual BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2522–2532, Online. 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.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Giuseppe Samo, Vivi Nastase, Chunyang Jiang, and Paola Merlo. 2023. BLM-s/lE: A structured dataset of English spray-load verb alternations for testing generalization in LLMs. In *Findings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Pedro Savarese, Hugo Silva, and Michael Maire. 2020. Winning the lottery with continuous sparsification. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin

Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *The Seventh International Conference on Learning Representations (ICLR)*, pages 235–249.

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.

Michael Wilson, Jackson Petty, and Robert Frank. 2023. How abstract is linguistic generalization in large language models? experiments with argument structure. *Transactions of the Association for Computational Linguistics*, 11:1377–1395.

David Yi, James Bruno, Jiayu Han, Peter Zukerman, and Shane Steinert-Threlkeld. 2022. Probing for understanding of English verb classes and alternations in large pre-trained language models. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 142–152, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Figure 9: Differences between value distributions in each CNN output node (i.e. each node corresponding to the application of the kernel from each channel on the sentence embeddings, according to the kernel size and stride) for sets of sentences with minimally different patters: (top) patterns differ in only one grammatical number attribute for one chunk, (bottom) patterns differ only in length. Each panel corresponds to one region of the sentence embedding the size of the kernel. The y-axis represents the channels of the CNN. The x-axis represents the latent units in different colours (the stronger the color, the higher the value, max = 1), and the pairs of compared patterns represented as adjacent rectangles. The difference between the patterns is written below the x-axis.