P.STAR: Parameter-free Structure Analysis via Reordering for Lossless Visualization of Layers in Language Models

Mennatallah El-Assady ♢ Antske Fokkens ♣ Pantea Haghighatkhah ♠ Tim Ophelders ♠♡
Pia Sommerauer ♣ Bettina Speckmann ♠ Kevin Verbeek ♠

♢ ETH Zürich ♣ VU Amsterdam ♠ TU Eindhoven ♡ Utrecht University

ABSTRACT

We introduce P.STAR: a new parameter-free method that can be used to inspect the embedding space of a set of instance representations (e.g. words in context) across layers of a contextualized language model at one glance and without loss of information. We use a matrix reordering approach that relies on signature distances; instances are represented in terms of their distances to all other instances in the set. The reordered matrix of signature representations provides a visual depiction of how structures emerge across layers of a contextualized model. We validate our approach by comparing how words in context are grouped to their PoS and Sense labels in two annotated datasets. In line with previous research, we observe that lower-layer representations emphasize PoS-information while higher layers group instances on the basis of senses. We also show that our reordering approach can identify errors in the labels.

1 INTRODUCTION

Despite progress in the field of interpretability, analyzing linguistic information in language models remains challenging. Layer-wise analyses of models can yield valuable theoretical and practical insights. Being able to observe whether and how linguistic structure emerges can increase our understanding of what models purely trained on textual data can learn about language. Some applications of large language models (LLM) benefit substantially from locating which layers of a model represent specific information (e.g. where to find information on word senses for semantic change detection).

We consider the embedding representations of different *instances*, that is, words in a specific context (e.g. sentence). Instances can be similar to or different from each other in different ways. For instance, words in specific contexts can be similar or different in terms of their syntactic or semantic properties. If a specific layer mainly captures syntactic information, we can expect that instances with highly similar syntactic properties will be relatively close when comparing their embedding representations from this layer. When studying the structure of the embedding space obtained from a layer mainly capturing semantic information, we may find the same instances to be (much) further apart.

In this paper, we want to analyze the structure of embedding representations in the different layers of a contextualized model. These embeddings can be represented as a set of points $\{x_1, \ldots, x_n\}$ in high-dimensional space. There are several ways to study the structure of such point sets. *Dimensionality reduction techniques* can be used to represent the points using low-dimensional (specifically, 2-dimensional) vectors, which can then be visualized to reveal the structure of the underlying embedding. However, such dimension reductions (especially to 2D) induce a significant loss of information, and the visualized structure may deviate strongly from the original structure in the embedding. Another technique to analyze the structure of embeddings or point sets is *clustering*, which aims to find groups of similar points. However, clustering techniques typically require the user to choose parameters, such as the number of clusters or a distance threshold. Especially problematic is that the embeddings in different layers of a contextual language model for the same data still often require different parameters to reveal the structure of the embeddings clearly.

We introduce P.STAR: a **parameter-free** method that reveals the structure of a contextual embedding **at a glance**, with **no loss of information**. We validate our approach through an analysis of the emergence of part-of-speech (PoS) information and word senses in a contextualized model, as previous research suggests that part-of-speech information tends to be captured in lower layers of a model, while sense information tends to be captured in higher layers of the model (see, for example, [15, 28]).

A central component of our method consists of a new distance measure: the *signature distance*. Instead of relying on cosine distances between embeddings directly, we create new embeddings that represent the cosine distances between an instance and all other instances from a dataset. This signature distance is loosely based on previous work from Lexical Semantic Change Detection [11, 20]. We are, to our knowledge, the first to use it to compare embeddings from instance representations produced by contextual models.

Our paper makes the following contributions: We propose a new, parameter-free method for analyzing the structure of embedding spaces using signature distances and matrix reordering (Section 3). We implemented our approach and show how the resulting matrices visualize embedding space structure at one glance (Section 4). We validate P.STAR by showing that the structures it identifies clearly correspond to information about PoS and word senses and demonstrate how it can be used to identify anomalies in data (Section 5).

2 RELATED WORK

Within the field of model analysis and interpretability, several methods for analyzing internal model representations have been proposed [4]. The earliest and still widely used approaches use probing. This paradigm relies on testing whether the prediction of a specific linguistic property can be learned from embedding representations using labeled data and a simple (often linear) classifier. Probing results have been shown to vary substantially depending on the choice of classifier; the brittle nature of probing results has raised questions about the reliability of this approach [3, 13].

Clustering approaches can be seen as more direct ways of representing structure and have been used to analyze and visualize hidden layer representations (se, for example, [1, 6]). While clustering approaches do not require a classifier, they still depend on crucial parameters (e.g. the number of clusters). Zhou and Srikumar [31] propose hierarchical clustering as a parameter-free alternative, however, their approach crucially relies on labeled data to form clusters.

Embedding and Matrix Visualizations. Huang *et al.* [14] recently surveyed work in the visualization community that attempts to find and explain structures in embedding spaces. Of particular interest are the *Embedding Comparator* by [5] comparing structure using projections, and *LMFingerprints* by [24] comparing embedding spaces based on scoring. Additionally, some work emphasizes layer projection alignment [25]. Lastly, another direction of research is to compare the effect of language model adaptors visually [23]. All of these approaches, however, reduce the dimension of the data shown

979-8-3315-8037-7/25/$31.00 ©2025 IEEE
DOI 10.1109/VIS4DH69385.2025.00013

and hence lose a significant amount of information.

On the topic of matrix-based visualisation and reordering techniques, the current state-of-the-art [29] shows that matrix orders computed via solutions to the Travelling Salesperson Problem optimize for *Moran's I*, a spatial auto-correlation metric, which captures all established patterns in relational matrices well. This fact is a further indication that optimal linear orders reveal meaningful structure in distance matrices and supports our chosen approach.

Embedding Distances and Similarities. The basic idea behind our approach is to reorder embeddings according to their similarity. Cosine distances are often used to determine the similarity between embeddings but are also known to be susceptible to noise [27]. Zhou *et al.* [30] also illustrate that the cosine of contextualized representations underestimates the similarity of their corresponding words if they have a high frequency in the training corpora. As mentioned in the introduction, we propose to use the *signature distance* instead. A similar approach can be found in research on semantic change [11, 20, 21]. Here, the signature distance of a word consists of the distance between the representation of a target word to all other words in the corpus. Though primarily introduced to address the alignment problem that occurs when aiming to compare embeddings created from different corpora, signature distances are also more robust to noise compared to first-order cosine distances (see Section 3). There is, however, a fundamental difference between the signature distances used for semantic change detection and the signatures we propose in this paper. In semantic change detection, the primary embeddings are static word embeddings that represent word meaning based on multiple occurrences in a corpus. Our primary embeddings are representations of a specific instance (e.g. a word in a sentence). We create second-order embeddings by taking the cosine distance of a specific instance to all other instances in our dataset. We create second-order embeddings for each layer. To our knowledge, we are the first to propose comparing instance representations produced by contextualized models with second-order representations.

3 Linear Ordering

As already discussed in the introduction, we aim to develop a method that reveals the structure of a contextual embedding and is parameter-free. For the first step of our method, we are inspired by the recent results of [19]. Their topological analysis of the structure of contextual embeddings reveals that its structure often mainly consists of linear parts. This suggests that it may be possible to order the data in a meaningful way. The goal is to order the data points such that similar points (that is, clusters) are close together in the order.

Let $\{x_1, \ldots, x_n\}$ be the (high-dimensional) points in a contextual embedding. The similarity between two points x_i and x_j is typically expressed using a distance measure $d(x_i, x_j)$, where the distance is smaller if the points are more similar. The goal is then to find a permutation (an ordering) $\{x_{\sigma(1)}, \ldots, x_{\sigma(n)}\}$ (where σ is a permutation of $\{1, \ldots, n\}$) such that if $d(x_i, x_j)$ is relatively small, then $|\sigma^{-1}(i) - \sigma^{-1}(j)|$ is also relatively small (since $\sigma^{-1}(i)$ is the position of x_i in the new order, this implies that x_i and x_j are close in the new order). We observe that this goal can be achieved by ensuring that the distances $d(x_{\sigma(i)}, x_{\sigma(i+1)})$ (for $1 \leq i < n$) are as small as possible. If we want to minimize the sum of these distances, then this problem is nearly equivalent to the well-known Traveling Salesperson Problem (TSP). The only difference is that for our problem, we do not need to loop back to the starting point. This variant of TSP is known as the Open (Loop) TSP problem (OTSP).

Finding this permutation still presents us with a challenge from a computational point of view since it is well-known that both TSP and the variant OTSP are NP-hard. However, as TSP is arguably the most well-known and most important combinatorial optimization problem that exists, many very efficient algorithms have been engineered that can solve this problem *optimally* for thousands of points in mere seconds (see Section 4 for more details). It is thus feasible to compute the desired linear ordering in a practical setting. Although most algorithms are specifically designed for the TSP problem, the OTSP problem can easily be modeled as a TSP problem by adding a dummy point x^* for which $d(x_i, x^*) = 0$ for all points x_i. The optimal OTSP order can then simply be obtained from the optimal TSP tour by removing the dummy point.

Signature distance. To define the linear ordering, we need to choose which distance measure d we use. A common distance measure in contextual embeddings is the cosine distance:

$$d_c(x_i, x_j) = 1 - \frac{x_i \cdot x_j}{\|x_i\| \|x_j\|},$$

where $x_i \cdot x_j$ is the dot product between the two vectors and $\|x_i\|$ is the Euclidean length of a vector. However, we believe that the cosine distance may not be the best distance measure for our method. Contextual language models try to learn context-dependent representations (vectors) of words in a specific context and with the correct relations (that is, distances) to vectors representing the word in other contexts. In more advanced Transformer models, these representations are not directly learned, but emerge as a byproduct and can be extracted. The vectors themselves are not relevant, but the relations between vector representations (distances, inner products) capture the context of a word. However, this implies that different vector representations may capture the same context, which can be problematic when comparing multiple instances of the same word. Consider the example in Figure 1: the top and the bottom embedding represent nearly identical relative distances; only the distance between the two blue points are different. However, with respect to their context (the black points) the two blue points should be considered very similar and hence have a low distance. In high-dimensional spaces this effect can be even larger, since there is more freedom for the model to choose good vectors. Similar effects have already been observed in the literature as well [27]. We believe that this problem may cause a significant amount of noise in the distances measured directly with the standard cosine distance.

We hence opt to use a distance measure that is more stable with respect to small differences in the vector representations. Specifically, we consider the *signature* of a point x_i, defined as follows:

$$S(x_i) = (d_c(x_1, x_i), d_c(x_2, x_i), \ldots, d_c(x_n, x_i)).$$

The signature $S(x_i)$ of a point x_i is an n-dimensional vector that represents the distances of x_i to all other points. As a result, $S(x_i)$ more directly represents the context of x_i with respect to the other points in the data. This representation avoids the problem sketched above since the signatures of two points with similar context (the two blue points in Figure 1) are very similar. We can then define the *signature distance* between two points as follows:

$$d_S(x_i, x_j) = \frac{1}{\sqrt{n}} \cdot \|S(x_i) - S(x_j)\|_2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm. We normalize by $\sqrt{n}$ so that the signature distance is independent from vector length. Overall, we believe that the signature distance and cosine distance are strongly

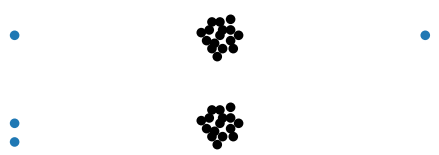


Figure 1: The relative distances represented by top and bottom embedding are nearly identical.

correlated, but that the signature distance is more consistent and less noisy than the cosine distance. The signature distance also tends to emphasize small distances and amplify larger distances, which is beneficial for our method.

To simplify notation in the remainder of this paper, we assume from now on that the points $\{x_1,\ldots,x_n\}$ in a contextual embedding are already given in the order of the optimal OTSP path using the signature distance.

4 Implementation and Visualization

In this section, we first discuss how the linear ordering described in Section 3 can be computed in practice. Then we show how the resulting ordering can be used to create a visualization that clearly reflects the underlying structure of the data at a glance.

Computing the order. As explained in Section 3, we can obtain the desired linear ordering of the instances by solving an OTSP problem. Arguably the most effective solver for TSP problems is Concorde[1] developed by Applegate *et al.* [2]. We specifically use the version of Concorde that is available on the NEOS (Network-Enabled Optimization System) Server[2] [8, 10, 12].

As input, Concorde requires a distance matrix M representing all the distances between the points in the data. We fill M using the signature distances $d_S(x_i,x_j)$ in $M(i,j)$. However, as already mentioned in Section 3 we additionally need to include a dummy point x^* that has distance 0 to all points. We thus add an additional row and column to M consisting only of zeroes. The resulting matrix M cannot be used directly since Concorde expects integer distances as input. We, therefore, compute integer distances by multiplying the distances with 10^3 (specifically, $M'(i,j) = \lfloor 10^3 \times M(i,j) \rfloor$).

Given the distance matrix as input, Concorde can compute the optimal TSP tour in mere seconds. The resulting tour still contains the dummy point x^*. To obtain the desired linear ordering, we simply scan through the tour to find the dummy point x^*. The following point in the tour is then the first point x_1 in the linear ordering, and we complete the ordering by following the points in the tour until we reach x^* again.

Visualization. We are now ready to visualize the structure of the underlying data using the linear ordering $\{x_1,\ldots,x_n\}$. Though we can explore our data and study the structural patterns that appear in our data in isolation, we can also use them in combination with annotated datasets to dive deeper into linguistic properties. Such annotated datasets additionally allow us to validate our method from a linguistic point of view. We hence additionally assume that we are given one or more tags a_i for each instance x_i that captures a particular linguistic property (see Section 5 for concrete examples). We assume that the tags a_i are in the form of categorical data, typically allowing only a small number of categories. Note that neither our method nor our visualization requires these tags; they are an optional feature that we include for both for validation and for deeper linguistic exploration.

A direct and complete matrix visualization of the distance matrix M between the instances, following the computed linear ordering, forms the core of our visualization. Specifically, we color the pixel at row i and column j according to the distance $d_S(x_i,x_j)$ in M. We map distance to color according to the following scheme:

Note that the mapping covers only distances in the range $[0, 0.6]$. This range was determined based on our data described in Section 5, where we explicitly excluded outliers from influencing the mapping. However, for other data a different range might be more suitable.

[1]The code and binaries can be found here.

[2]The NEOS Server is a free service that provides access to state-of-the-art solvers for numerical optimization problems. The solvers run on distributed high-performance machines, enabling them to handle large inputs.

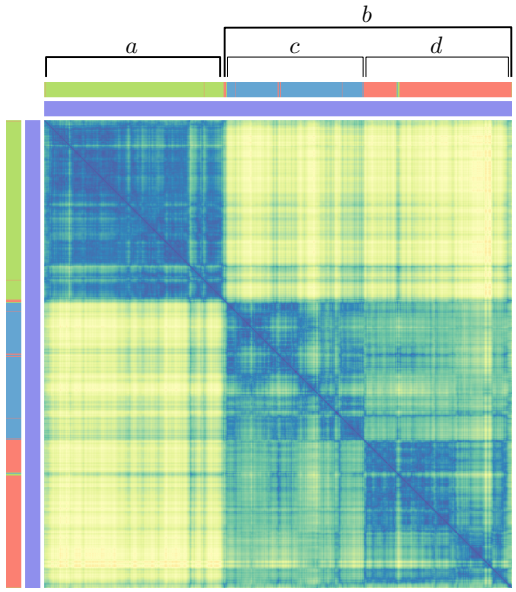


Figure 2: Matrix visualization of a set of instances. Note that the embeddings with the same tag form groups marked by the brackets. There are two prominent clusters a and b. Cluster b contains two subclusters c and d, where each subcluster has a different tag. You can observe the global structure of the instances and the relative distances between clusters.

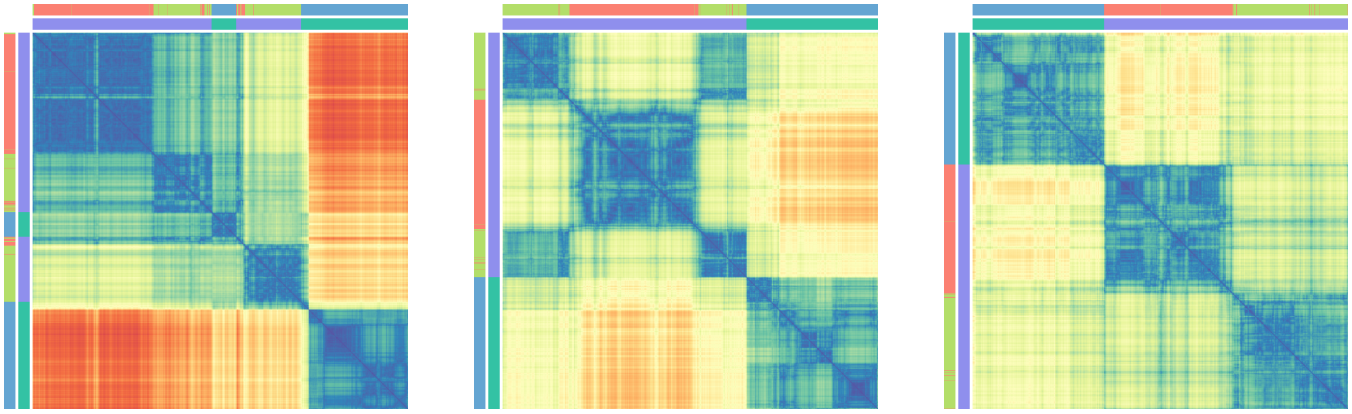
An example of a matrix visualization is shown in Figure 2. This visualization should be interpreted as follows. If we look along the diagonal of the matrix, we see a number of blue squares with differing shades of blue, as indicated by a, c, and d. These blue squares indicate clusters in the data, and the specific shade of blue indicates how close the instances in the cluster are to each other. If we look more closely at Figure 2, then we can see that c and d together form a larger cluster b that is still well-separated from the cluster a. Thanks to the linear ordering and the matrix visualization, we are able to see clusters at different levels in one picture.

To visualize the tags of the instances, we further add bars on top and to the left of the matrix visualization, where we use one bar per tag. Each bar contains a short line per instance that is colored according to the category of the respective instance in the same linear ordering. In Figure 2, we can see a visualization that includes two tags, although one tag is the same for all instances (the inner bar). The outer bar shows that the linear ordering does not only help to expose the clusters in the underlying data but is also able to nearly order the instances according to the linguistic property represented by the tag. Note that the tags are not used when computing the linear ordering, so this directly implies that the linguistic information represented by the tag is encoded well in the semantic embedding.

5 Linguistic Validation

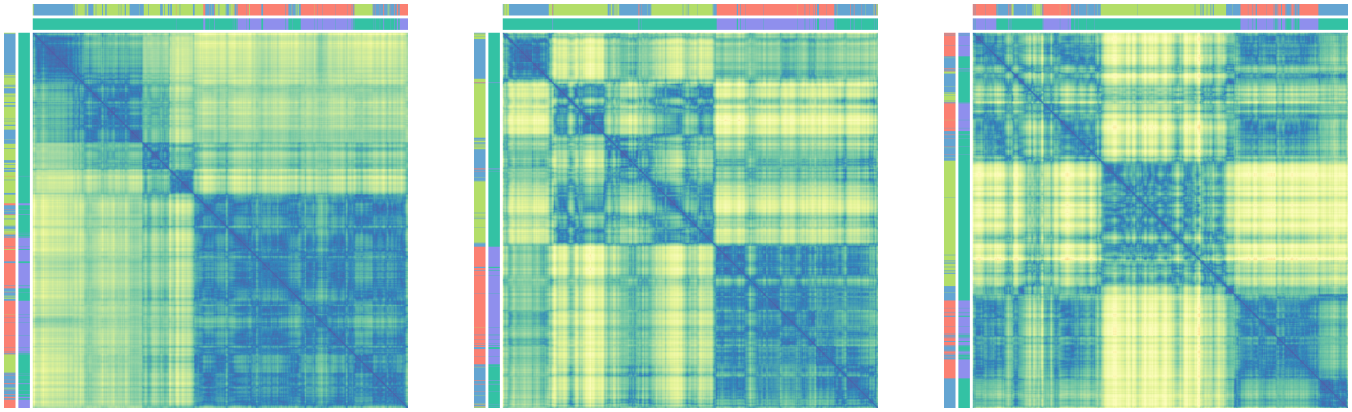
In this section, we provide a validation of our method. We show how information about part of speech and word senses emerges on different layers of a model and compare the resulting structures to part-of-speech (PoS) and sense-annotated data.

Models. We use the `BERT-base-uncased` model [9] and GPT2 [18] to obtain the embeddings for the words in our selection. To obtain the embeddings of a word w, for every instance of the word in context c, we tokenize c and feed the tokens to the model. `BERT-base-uncased` and `GPT2` both have 12 layers; since we want



Layer 3 Layer 7 Layer 10

Figure 3: P.STAR applied to Layers 3, 7, and 10 of a BERT embedding of **state**. Senses and PoS:
the territory occupied by one of the constituent administrative districts of a nation [noun]
the way something is with respect to its main attributes [noun]
express in words [verb]



Layer 3 Layer 7 Layer 9

Figure 4: P.STAR applied to Layers 3, 7, and 9 of a BERT embedding of **work**. Senses and PoS:
perform as expected when applied [verb]
the occupation for which you are paid [noun]
be employed [verb]

to analyze the structure of every layer separately, we collect the output vectors (hidden states) associated with the word w from every layer of the model. Note that none of the words we have selected are broken into subwords (tokens) by the BERT or GPT2 tokenizer, hence every instance is a complete word in context.

Data. We are using two English sense annotated datasets, SemCor3.0 [17] and OMSTI [26].[3] Both datasets are widely used for the Word Sense Disambiguation (WSD) task. Semcor3.0 is manually annotated with WordNet3.0 senses and contains a total of 226,040 sense annotations. The OMSTI (One Million Sense-Tagged Instances) dataset is automatically annotated using an alignment-based WSD approach [7]. Both datasets contain PoS tags. We select polysemous words that can serve to illustrate different degrees and types of polysemy; in some cases, word senses are not closely related to each other and should thus be neatly separable based on their contexts. In other cases, senses can be closely related; they may not always be clearly distinguishable based on context. We extract all sentences containing our selected words (and senses). The NEOS server can handle files upto a size of 16.7 MB. If words with many instances exceed this size, we take a random sample of equal size for each sense.

Matrix creation. For each word, we build a matrix representing all its instances (i.e. mentions in sentences in the annotated corpora). We build one matrix for each layer of the model. As the distances between the instances of a word will vary across the layers of the models, instances will be ordered differently from one layer to another. Our ordering approach thus directly reflects how the structure of instance representations changes from one layer to another. The linguistic information we are validating against is represented by bars: the inner bar represents PoS information, and the outer bar represents different senses.

Analysis. We showcase the insights provided by P.STAR using three examples. The first two examples demonstrate how our reordering approach highlights how different aspects of linguistic information emerge on different layers of a model. We select the two words *state* and *work*. Both words have nominal and verbal uses as well as multiple senses. While *state* should result in clearly separated senses,

[3] `http://lcl.uniroma1.it/wsdeval/training-data`

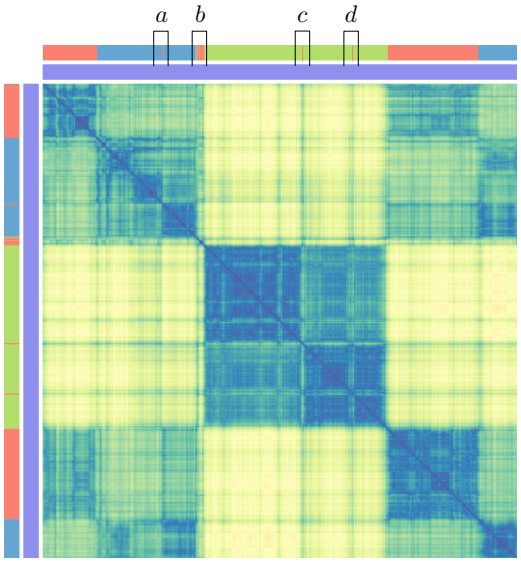


Figure 5: P.STAR applied to Layer 12 of a BERT embedding of **cell**. Senses and PoS:
a room where a prisoner is kept [noun]
(biology) the basic structural and functional unit of all organisms [noun]
a device that delivers an electric current as the result of a chemical reaction [noun]
The markers *a*, *b*, *c* and *d* indicate sense labeling errors.

the senses of *work* are closely related and are not expected to result in clear clusters. For both words, we expect clear separations of their nominal and verbal uses in the early layers. We show the first two examples across multiple layers in Figure 3 (*state*), Figure 4 (*work*) and one layer for *cell* in Figure 5. Here we discuss the matrices created for `BERT` representations; those computed with `GPT-2` show similar tendencies.

For *state*, we analyze two nominal senses ("territory" and "condition") and its verbal sense of "to express". All three senses are distinct and should result in clear separations in higher layers. The distinction between nominal and verbal use should become apparent in lower layers. Figure 3 confirms our expectation; the inner bar (in turquoise and purple) reveals that instances of the verb *state* start to form two groups in Layer 3 and are neatly separated into verbal and nominal uses in Layer 7. In the same layer, we see that the instances *state* in the sense of condition appears between two groups of instances of *state* in the sense of territory (see the outer bar: green and orange labels intertwine).

We expect a more complex scenario for the word *work*. We focus on one nominal and two verbal senses; "occupation" (noun), "to be employed" (verb), and "to perform as expected when applied; to function" (verb). The verbal sense of "to be employed" is relatively close to the nominal sense of "occupation". We expect the lower layers to reflect a separation into nominal and verbal uses but the higher layers to reflect the similarity between the two closely related senses across PoS boundaries. We can see that P.STAR reflects this scenario in Figure 4. We observe that instances of *work* used as a noun are mostly separated from its verbal use in Layer 7. Instances with the related verbal sense of *work* are partially placed among the nominal occurrences and partially mixed with the other verbal use of *work* (in the sense of functioning). In Layer 9, more instances of *work* in the sense of functioning are ordered next to each other. The related verbal and nominal instances referring to occupation are more mixed compared to Layer 7, revealing that semantic similarity increased in importance compared to syntactic similarity when moving from Layer 7 to Layer 9.

Our third example shows how P.STAR can be used to gain new insights into labeled data. Figure 5 is a matrix visualization of embeddings from the 12th layer of the noun *cell* labeled as three different senses: *A prison cell*, *biological cell* and *a device that delivers an electric current, a battery cell*. We observe multiple anomalies in the sense bar, marked as *a*, *b*, *c* and *d*. We inspected the corresponding instances. First, *a* highlights instances that are labeled (biological cell) but placed in between instances corresponding to (battery). Inspection shows that these are mislabelled instances of OMSTI (e.g. *"Membrane cells release less hazardous substances and are more energy-efficient than the older techniques (e.g. KEMI, 2004)."*). When inspecting the instances in *b*, *c*, and *d*, we also observe that *cell* does not refer to a biological cell. In these instances, *cell* refers to a group of people, e.g. a terrorist cell. The placement of *c* and *d*, among references to prison cells, can be explained by the fact that those instances are semantically related to scenarios that involve prisons: *"[...] terrorists are being arrested, their cells are being broken up, [...]."*

Our first two analyses show that P.STAR reveals how word-in-context representations group instances according to PoS information in their lower layers and regroup them in higher layers according to semantic information. Our third inspection also demonstrates how P.STAR can be used to find annotation errors; in particular, we identified wrong sense tags in the automatically labeled OMSTI corpus by inspecting instances that were placed in the surrounding of instances with different labels.

6 Discussion

Our method relies on the visual inspection of matrices; as such, our validation consists of a comparatively small set of examples that serve as a first illustration. The clarity of the observations does, however, provide convincing evidence for the reliability of our method. This first step opens up new directions for exploration. One of the reasons why the approach provides such clear structures is that we make use of signature distances. The second-order distance approach is more robust than cosine distances between embeddings.

The major advantage of our method is that it can represent an entire set of instances at one glance without reducing dimensions, or setting parameters that might influence results. As such, it can provide a more direct depiction of structures than could be achieved through clustering approaches. As demonstrated in our validation, our method shows how clusters and nested clusters arise from the linear reordering of second-order signature representations.

From a linguistic analysis point of view, we believe that P.STAR could offer a powerful new method for the analysis of linguistic properties in contextualized language models and could yield new theoretical insights. From a practical perspective, P.STAR can be used to detect layers in contextualized models that are likely to carry useful information for a particular task or for further model fine-tuning. Our analysis shows that layers that capture word senses particularly well can be identified. Word sense information is of central importance for the task of Lexical Semantic Change Detection. Both the use of signature distances and better pinpointing where sense is represented most can help boost the performance of contextualized models for a task where static word embeddings often still outperform them [16, 22].

In the future, we will build on our P.STAR method to develop interactive visual analytics tools, enabling the exploration of patterns across model layers, for example, to investigate the progression of the learned semantic separability across the layers.

REFERENCES

[1] A. Alishahi, M. Barking, and G. Chrupała. Encoding of phonology in a recurrent neural model of grounded speech. *CoNLL 2017*, p. 368, 2017.

[2] D. Applegate, R. Bixby, W. Cook, and V. Chvátal. On the solution of traveling salesman problems. *Proceedings of the International Congress of Mathematicians*, 1998.

[3] Y. Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022. doi: 10.1162/coli_a_00422

[4] Y. Belinkov and J. Glass. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019.

[5] A. Boggust, B. Carter, and A. Satyanarayan. Embedding comparator: Visualizing differences in global structure and local neighborhoods via small multiples. In *27th International Conference on Intelligent User Interfaces*, p. 746–766, 2022. doi: 10.1145/3490099.3511122

[6] G. Brunner, Y. Wang, R. Wattenhofer, and M. Weigelt. Natural language multitasking: analyzing and improving syntactic saliency of hidden representations. . *The 31st Annual Conference on Neural Information Processing (NIPS)—Workshop on Learning Disentangled Features: From Perception to Control*, 2017.

[7] Y. S. Chan and H. T. Ng. Scaling up word sense disambiguation via parallel texts. In *Proceedings of the 20th National Conference on Artificial Intelligence*, vol. 3, p. 1037–1042. AAAI Press, 2005.

[8] J. Czyzyk, M. P. Mesnier, and J. J. Moré. The neos server. *IEEE Journal on Computational Science and Engineering*, 5(3):68–75, 1998.

[9] 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)*, pp. 4171–4186, 2019. doi: 10.18653/v1/N19-1423

[10] E. D. Dolan. The neos server 4.0 administrative guide. Technical Memorandum ANL/MCS-TM-250, Mathematics and Computer Science Division, Argonne National Laboratory, 2001.

[11] S. Eger and A. Mehler. On the linearity of semantic change: Investigating meaning variation via dynamic graph models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 52–58, 2016. doi: 10.18653/v1/P16-2009

[12] W. Gropp and J. J. Moré. Optimization environments and the neos server. In M. D. Buhman and A. Iserles, eds., *Approximation Theory and Optimization*, pp. 167 – 182. Cambridge University Press, 1997.

[13] J. Hewitt and C. D. Manning. 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)*, pp. 4129–4138, 2019. doi: 10.18653/v1/N19-1419

[14] Z. Huang, D. Witschard, K. Kucher, and A. Kerren. VA + Embeddings STAR: A State-of-the-Art Report on the Use of Embeddings in Visual Analytics. *Computer Graphics Forum*, 2023. doi: 10.1111/cgf.14859

[15] G. Jawahar, B. Sagot, and D. Seddah. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3651–3657, 2019. doi: 10.18653/v1/P19-1356

[16] A. Kutuzov, E. Velldal, and L. Øvrelid. Contextualized embeddings for semantic change detection: Lessons learned. In *Northern European Journal of Language Technology, Volume 8*, 2022. doi: 10.3384/nejlt.2000-1533.2022.3478

[17] G. A. Miller, M. Chodorow, S. Landes, C. Leacock, and R. G. Thomas. Using a semantic concordance for sense identification. In *Human Language Technology: Proceedings of a Workshop*, pp. 8–11, 1994.

[18] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[19] A. Rathore, Y. Zhou, V. Srikumar, and B. Wang. Topobert: Exploring the topology of fine-tuned word representations. *Information Visualization*, 22(3):186–208, 2023.

[20] B. B. Rieger. Semiotic cognitive information processing: Learning to understand discourse. a systemic model of meaning constitution. *Adaptivity and Learning: An Interdisciplinary Debate*, pp. 347–403, 2003.

[21] M. A. Rodda, M. S. Senaldi, and A. Lenci. Panta rei: Tracking semantic change with distributional semantics in ancient greek. *IJCoL. Italian Journal of Computational Linguistics*, 3(3-1):11–24, 2017.

[22] D. Schlechtweg, B. McGillivray, S. Hengchen, H. Dubossarsky, and N. Tahmasebi. SemEval-2020 task 1: Unsupervised lexical semantic change detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pp. 1–23, 2020. doi: 10.18653/v1/2020.semeval-1.1

[23] R. Sevastjanova, E. Cakmak, S. Ravfogel, R. Cotterell, and M. El-Assady. Visual comparison of language model adaptation. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1178–1188, 2022.

[24] R. Sevastjanova, A. Kalouli, C. Beck, H. Hauptmann, and M. El-Assady. Lmfingerprints: Visual explanations of language model embedding spaces through layerwise contextualization scores. In *Eurographics Conference on Visualization (EuroVis)*, vol. 41, pp. 295–307, 2022. doi: 10.1111/cgf.14541

[25] R. Sevastjanova, A.-L. Kalouli, C. Beck, H. Schäfer, M. El-Assady, C. Zong, F. Xia, W. Li, and R. Navigli. Explaining contextualization in language models using visual analytics. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, vol. 1: Long Papers, pp. 464–476, 2021. doi: 10.18653/v1/2021.acl-long.39

[26] K. Taghipour and H. T. Ng. One million sense-tagged instances for word sense disambiguation and induction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pp. 338–344, 2015. doi: 10.18653/v1/K15-1037

[27] N. Tahmasebi, L. Borin, and A. Jatowt. Survey of computational approaches to lexical semantic change detection. *Computational approaches to semantic change*, 6:1, 2021.

[28] I. Tenney, D. Das, and E. Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, 2019. doi: 10.18653/v1/P19-1452

[29] N. van Beusekom, W. Meulemans, and B. Speckmann. Simultaneous matrix orderings for graph collections. *IEEE Trans. Vis. Comput. Graph.*, 28(1):1–10, 2022. doi: 10.1109/TVCG.2021.3114773

[30] K. Zhou, K. Ethayarajh, D. Card, and D. Jurafsky. Problems with cosine as a measure of embedding similarity for high frequency words. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, vol. 2, pp. 401–423, 2022. doi: 10.18653/v1/2022.acl-short.45

[31] Y. Zhou and V. Srikumar. DirectProbe: Studying representations without classifiers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5070–5083, 2021. doi: 10.18653/v1/2021.naacl-main.401