Scaling up Discovery of Latent Concepts in Deep NLP Models

Majd Hawasly **Fahim Dalvi** **Nadir Durrani**
Qatar Computing Research Institute, HBKU, Doha, Qatar
{mhawasly,faimaduddin,ndurrani}@hbku.edu.qa

Abstract

Despite the revolution caused by deep NLP models, they remain black boxes, necessitating research to understand their decision-making processes. A recent work by Dalvi et al. (2022) carried out representation analysis through the lens of clustering latent spaces within pre-trained models (PLMs), but that approach is limited to small scale due to the high cost of running Agglomerative hierarchical clustering. This paper studies clustering algorithms in order to scale the discovery of encoded concepts in PLM representations to larger datasets and models. We propose metrics for assessing the quality of discovered latent concepts and use them to compare the studied clustering algorithms. We found that K-Means-based concept discovery significantly enhances efficiency while maintaining the quality of the obtained concepts. Furthermore, we demonstrate the practicality of this newfound efficiency by scaling latent concept discovery to LLMs and phrasal concepts.[1]

1 Introduction

Transformer-based language models excel at revealing intricate patterns, semantic relationships, and nuanced linguistic dependencies concealed within vast textual datasets through unsupervised learning. Their capability to encode complex abstractions, surpassing surface-level word meanings, has resulted in significant advancements across various natural language understanding tasks. A considerable body of research dedicated to interpreting pre-trained language models (e.g. Durrani et al. (2019); Tenney et al. (2019); Geva et al. (2021); Sajjad et al. (2022a) among others) seeks to answer the question: *What knowledge is learned within these models?* Researchers have delved into the concepts encoded in pre-trained language models by probing them against various linguistic properties. Our work facilitates this line of work in interpretability by scaling up discovery of latent concepts learned within pre-trained language models.

Mikolov et al. (2013) demonstrated that words exhibit a tendency to form clusters in high-dimensional spaces, reflecting their morphological, syntactic, and semantic relationships. Building upon this foundational insight, recent studies (Michael et al., 2020; Dalvi et al., 2022; Fu and Lapata, 2022) delve into representation analysis by exploring latent spaces within pre-trained models. Dalvi et al. (2022) discovered encoded concepts in pre-trained models by employing Agglomerative hierarchical clustering (Gowda and Krishna, 1978) on the contextualized representations in the BERT model (Devlin et al., 2019). However, a fundamental limitation of their work is the computational expense of the underlying methodology. Since contextual representations are high-dimensional, only a limited amount of data can be clustered to extract the latent concepts. This significantly undermines the purpose of concept discovery, providing only a limited perspective on the spectrum of concepts that might be learned within the model and significantly limiting the scalability of the approach.

In this work, we aim to address this shortcoming by employing computationally-cheaper clustering algorithms, specifically comparing three algorithms in quality and computational efficiency: *Agglomerative Hierarchical Clustering*, *Leaders Algorithm*, and *K-Means Clustering*. As there is no inherent groundtruth clustering we can rely on to measure the quality of the algorithms in the latent space, we introduce a metric with two dimensions, *alignment* and *coverage*, to measure the "goodness" of a clustering. We also show that scaling the underlying data for concept discovery results in significantly better results, as well as enables new directions that were previously unexplored. In summary we make the following contributions:

[1]Source Code: https://github.com/qcri/Latent_Concept_Analysis

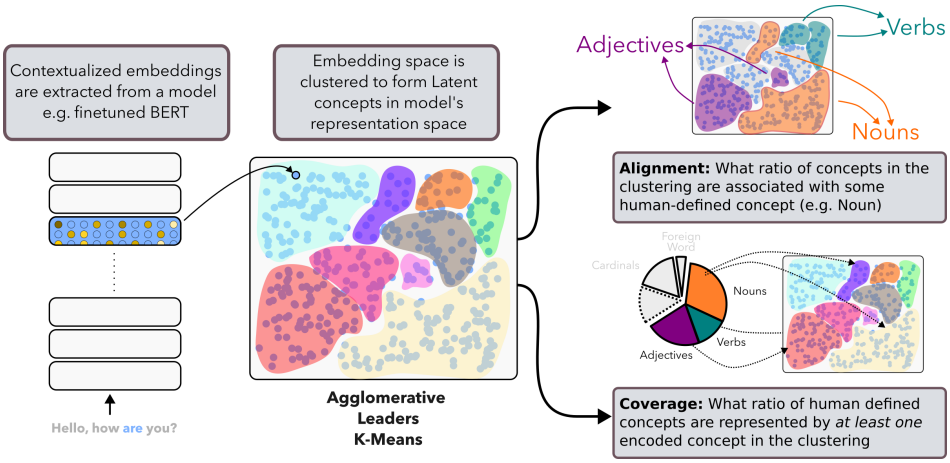


Figure 1: Discovery of encoded concepts within a PLM using clustering of contextualized embeddings, and evaluation of discovered concepts through *alignment* and *coverage* metrics with respect to human ontologies.

- We present a comprehensive comparison between various clustering techniques regarding their quality and efficiency for the task of latent concept discovery.
- We introduce a metric with two dimensions to measure the quality of extracted latent concepts: *alignment* and *coverage* of linguistic ontologies.
- We demonstrate that K-Means exhibits the capacity to handle vast datasets effectively while still producing latent concepts of roughly the same quality as Agglomerative hierarchical clustering.
- We show that increasing the size of the dataset used for clustering leads to higher-quality concept discovery, improving the coverage of POS tags by 8% on average in the last layer of fine-tuned BERT models, and 26% in the base Llama2 model.
- We present preliminary results in two new directions that K-Means affords us: exploration of latent concepts at a level higher than just words (e.g. phrases), and scaling concept discovery to large language models (LLMs).

2 Concept Discovery

Our investigation builds upon the work of discovering Latent Ontologies in contextualized representations (Dalvi et al., 2022). At a high level, feature vectors (contextualized representations) are initially generated by performing a forward pass on a pre-trained language model. The representations are then clustered to uncover the encoded concepts of the model (See Figure 1 for illustration). A concept, in this context, can be understood as a collection of words grouped together based on some linguistic relationship, such as lexical, semantic, syntactic, or morphological connections. Figure 2 showcases concepts within the latent space of the BERT model, wherein word representations are arranged based on distinct linguistic concepts.

Formally, consider a pre-trained model $\mathbf{M}$ with L layers: $l_1, l_2, \ldots, l_L$. Using a dataset of S sentences totaling N tokens, $\mathcal{D} = [w_1, w_2, \ldots, w_N]$, we generate feature vectors: $\mathcal{D} \xrightarrow{\mathbf{M}_l} \mathbf{z}^l = [\mathbf{z}^l_1, \ldots, \mathbf{z}^l_N]$, where $\mathbf{z}^l_i$ is the contextualized representation for the word w_i within the context of its sentence at layer l. A clustering algorithm is then employed in the per-layer feature vector space $\mathbf{z}^l$ to discover layer-l encoded concepts.

2.1 Clustering algorithms

In this paper, our focus is to increase the scalability of latent concept discovery. Hence, we evaluate different clustering algorithms in order to find one that can produce similar or better categorization of the latent space of a model with higher computational efficiency than the originally proposed method. To this end, we study three algorithms:

Agglomerative hierarchical clustering

Dalvi et al. (2022) utilized Agglomerative hierarchical clustering to organize words. This clustering technique generates a binary tree with N leaves which represent singletons (clusters of individual data points/words). Conversely, all other nodes in the tree signify clusters formed by merging the

(a) Lexical: *ex-* (b) Morphological: Adjective (c) Syntactic: Verb (d) Semantic: Person

Figure 2: Examples of encoded concepts in BERT aligned with human-defined ontologies

members of their respective child nodes. The merging of clusters takes place iteratively, driven by Ward's minimum variance criterion which utilizes intra-cluster variance as a measure of dissimilarity. The similarity between vector representations is evaluated using Euclidean distance. To extract a total of K clusters from this hierarchical structure, the tree is cut at layer $N - K$, followed by the retrieval of nodes without parents. For instance, a cut at layer 0 yields N clusters, each comprising a single point, while a cut at layer $N - 1$ results in a solitary cluster containing all N points.

In terms of computational complexity, Agglomerative clustering exhibits a time complexity of $O(N^2(D + \log(N)))$ and a space complexity of $O(N^2)$ (Aggarwal et al., 2015). Here, D represents the dimensionality of the data points, as the method necessitates maintaining a distance matrix that is updated throughout the algorithm's execution. The quadratic complexity constraint in N confines the applicability of this method to small word datasets.

Leaders Algorithm

An effective strategy for enhancing the efficiency of Agglomerative hierarchical clustering involves preprocessing the data points in order to reduce their count from N to a much smaller value, denoted as $M \ll N$. The *Leaders Algorithm* (Hartigan, 1975) accomplishes this by making a single pass over the data in an arbitrary order. During this pass, any data point that lies within a distance of τ from a previously encountered point is classified as a *follower* to the former point which becomes a *leader*. Following this pass, each leader clique of connected points is condensed, often through the computation of a centroid. The resulting reduced dataset of centroids is then clustered, e.g. using Agglomerative hierarchical clustering. It is important to emphasize that the clustering outcome achieved through this method is not equivalent to directly applying Agglomerative hierarchical clustering on the original data. The results are contingent on both the arbitrary order established during the single pass and the chosen threshold value τ.

For this approach, the space complexity is $O(M^2)$, and while the clustering phase has the reduced time complexity of $O(M^2(D + \log(M)))$, the dominant factor influencing the time complexity of this algorithm is the $O(N^2)$ time complexity of the preprocessing phase.

K-Means clustering

K-Means clustering is a widely used machine learning technique for partitioning a dataset into distinct groups or clusters. The objective of K-Means is to group similar data points together while maximizing the dissimilarity between different clusters. It operates by proposing a set of centroids then iteratively assigning data points to the nearest cluster centroid and recalculating the centroids based on the newly-formed clusters. As the algorithm progresses, the clusters gradually represent coherent patterns or structures within the dataset. This process continues until convergence when the centroids stop updating or a maximum number of iterations is reached.

Given its susceptibility to getting trapped in local minima, the approach often incorporates random restarts to significantly enhance the exploration in the optimization space. Remarkably efficient, the K-Means clustering algorithm boasts a space complexity of $O(N(D + K))$, contingent on the dimensionality D of the data points and the number of clusters K (Jin and Han, 2010). For a sequence of I iterations, its time complexity stands at $O(NKDI)$ (Aggarwal et al., 2015).

3 Assessing Quality of Concept Discovery

With the unsupervised identification of *encoded concepts* by a clustering algorithm, a question arises: *how can we effectively compare various clustering algorithms in relation to their ability to uncover these concepts?* We introduce a measure that evaluates the alignment of encoded concepts in

light of linguistic ontologies (e.g. parts-of-speech tagging). Previous research (Kovaleva et al., 2019; Merchant et al., 2020; Durrani et al., 2021, 2022) showed that higher layers of PLMs get optimized for the task that the PLM is trained for, and that tuning a PLM for any task results in its latent space being skewed towards the output classes of the target task.[2] We use this finding to compare the concepts discovered via different algorithms, by measuring the alignment between the encoded concepts learned by a fine-tuned model to the human-defined concepts of the underlying task. Although we here employ a fine-tuned model to assess quality, it is crucial to emphasize that this is solely for evaluation purposes. The selected algorithm can subsequently be applied to any generic pre-trained language model to uncover its latent concepts as well.

Formally, consider a downstream task (e.g. POS tagging (Marcus et al., 1993)), for which we possess true class annotations for the input data $\mathcal{D}$ (i.e., per-word POS labels). For each tag, we construct a *human-defined concept* using the annotated data. For instance, $C_h(\texttt{VBD}) = \{died, smiled, explored, \dots\}$ defines a human concept comprising past-tense verbs, while $C_h(\texttt{NNS}) = \{boys, girls, rackets, \dots\}$ outlines a concept of plural nouns. Let $\mathcal{C}_\mathcal{H} = \{C_{h_1}, C_{h_2}, \dots, C_{h_n}\}$ be the set of all human-defined concepts for the task, and $\mathcal{C}_\mathcal{E} = \{C_{e_1}, C_{e_2}, \dots, C_{e_m}\}$ be the set of discovered encoded concepts within the latent space of the fine-tuned PLM. We define their **θ-alignment** as a function $\lambda_\theta(\mathcal{E}, \mathcal{H})$:

$$\lambda_\theta(\mathcal{E}, \mathcal{H}) = \frac{1}{2}\frac{\sum_\mathcal{E} \alpha_\theta(C_e)}{|\mathcal{C}_\mathcal{E}|} + \frac{1}{2}\frac{\sum_\mathcal{H} \kappa_\theta(C_h)}{|\mathcal{C}_\mathcal{H}|}, \text{ where}$$

$$\alpha_\theta(C_e) = \begin{cases} 1, & \text{if } \exists C_h \in \mathcal{C}_\mathcal{H} : \frac{|C_e \cap C_h|}{|C_e|} \geq \theta \\ 0, & \text{otherwise} \end{cases}$$

$$\kappa_\theta(C_h) = \begin{cases} 1, & \text{if } \exists C_e \in \mathcal{C}_\mathcal{E} : \frac{|C_e \cap C_h|}{|C_e|} \geq \theta \\ 0, & \text{otherwise} \end{cases}$$

The first term computes the ratio of discovered concepts that are aligned (up to $\theta \in [0, 1]$) to the human-defined concepts (*alignment*), while the second measures how many unique concepts within the human-defined ontology were recovered within the latent space (*coverage*). This latter term demarcates our metric from that of Dalvi et al. (2022), and we use a high threshold $\theta = 0.95$ in our experiments. Figure 1 shows a visual representation of the two terms.

[2] We verified this through our experiment, which involved comparing latent concepts before and after fine-tuning, as detailed in Section 4.2 and Figure 3.

4 Experimental Setup

4.1 Models and Tasks

We conducted experiments with three widely used transformer architectures: BERT-base-cased (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), and XLM-RoBERTa (Conneau et al., 2020), employing their base versions (comprising 13 layers and 768 dimensions). For our investigation of clustering quality, we fine-tuned the base models on conventional tasks that encompass fundamental linguistic concepts. These tasks included 1) morphological analysis using part-of-speech tagging with the Penn TreeBank dataset (Marcus et al., 1993); 2) syntax comprehension using CCG super tagging with the CCG TreeBank (Hockenmaier, 2006); and 3) semantic tagging using the Parallel Meaning Bank dataset (Abzianidze et al., 2017). Appendix A and B provide details of these datasets and the fine-tuning setup. We also employ two versions of Llama-2 (Touvron et al., 2023) for our experiments on large language models, specifically `Llama2-7B` and `Llama-2-7B-chat`, with an architecture of 32 layers and 10,000 embedding dimensions.

4.2 Calibrating latent space towards a task

PLMs are trained towards the generic task of language modeling through next word prediction. Following the pre-training phase, the model can be fine-tuned with additional training using a more specific annotated dataset tailored to a particular task. Our approach involves fine-tuning PLMs for downstream tasks with a human ontology to align the latent space, and then aligning the discovered encoded concepts with the target output labels. This allows us to use the θ-alignment function we described in Section 3 for clustering quality. To quantify the extent of this transformation, we evaluate the degree of overlap between the concepts encoded within the same layer of both base models and their fine-tuned counterparts to the human concepts. As depicted in Figure 3, the number of aligned concepts increase substantially in the upper layers of the fine-tuned models compared to base models. We found this to be true for all tasks and clustering algorithms undertaken in this study.

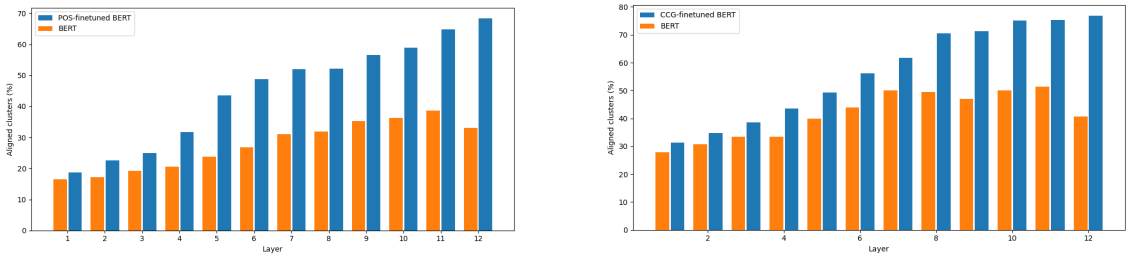


Figure 3: Alignment (percentage of discovered encoded concepts) of K-Means for POS (left) and CCG (right) in the base BERT model versus the corresponding fine-tuned models. The number of aligned concepts appreciate significantly in the higher layers of the tuned model in both cases.

4.3 Clustering

To generate data for clustering, we perform a forward-pass through the models on their respective training sets to generate contextualized feature vectors.[3] Subsequently, we applied various clustering algorithms to these vectors. This process was carried out independently for each layer, resulting in the generation of K clusters (i.e. encoded concepts) per layer. For our experiments, we set $K = 600$ as Dalvi et al. (2022) found that a K within the range of 600 to 1000 achieved a satisfactory balance between overly-extensive and inadequate clustering, while their exploration of other methods, such as ELbow and Silhouette, did not yield consistent outcomes. Note that K here does not have to correspond to the number of classes for the target task, as each target class may further be divided into sub-classes representing different facets (e.g. Adverbs can further be split into Adverbs of time, manner, place, etc.) as found by Mousi et al. (2023).

For Agglomerative hierarchical clustering, we used the implementation of `scikit-learn` (Pedregosa et al., 2011) (version 0.24.2) with euclidean distance and Ward linkage criterion. For leaders, we perform binary search to find the right threshold τ in order to reduce the dataset to the desired size for a computation budget before applying Agglomerative clustering. For efficiency, the single pass that compresses the data was implemented using `Annoy` approximate neighbor library.[4] For K-Means, we also use the standard `KMeans` implementation of `scikit-learn` with sampled initial seeds and 10 restarts.

[3]We use NeuroX toolkit (Dalvi et al., 2019).

[4]https://github.com/spotify/annoy

4.4 Alignment Threshold

We consider an encoded concept/cluster to be aligned with a human-defined concept when it exhibits an alignment of at least 95% in the number of words ($\theta = 0.95$), i.e. 95% of the words in the cluster belong to the human-defined concept and allowing for only a 5% margin of noise. Nonetheless, our patterns remain consistent for lower or higher thresholds. We only consider concepts that have more than 5 unique word-types. Note that the encoded concepts are based on contextualized embedding, i.e. the same word can have different embeddings depending on the context it appears in.

5 Results

In the following subsections, we present our comparison of the three clustering algorithms in various data and model regimes to identify the strengths and weaknesses of the underlying methods.

5.1 Concept discovery quality

We first compare the algorithms on exactly the same underlying dataset to answer the question *how does concept discovery quality compare across clustering algorithms?* Table 1 shows the metrics for three tasks and the three clustering algorithms for layer 12 embeddings of fine-tuned BERT-base-cased models. Since we are directly comparing the algorithms for quality, we use a subset of the data that all three methods are capable of processing. Note that the *Leaders* variant uses *Agglomerative* clustering after reducing the data to a manageable size, rendering it equivalent to Agglomerative clustering in this case.

In our results the *K-Means* algorithm demonstrates a superior performance over Agglomerative clustering in alignment and coverage when using the same data. In an attempt to investigate this further, we plot the distribution of the sizes

	Clustering	Size	Layer 12		
			Align. %	Cov. %	$\lambda_\theta(\mathcal{E},\mathcal{H})$
pos	Agglomerative	245K	47.8	**60.0**	0.54
	Leaders	245K	47.8	**60.0**	0.54
	K-Means	245K	**60.2**	**60.0**	**0.60**
ccg	Agglomerative	222K	67.2	22.6	0.45
	Leaders	222K	67.2	22.6	0.45
	K-Means	222K	**70.5**	**23.6**	**0.47**
sem	Agglomerative	223K	46.8	50.7	0.49
	Leaders	223K	46.8	50.7	0.49
	K-Means	223K	**58.8**	**59.7**	**0.59**

Table 1: Comparing aligned clusters and concepts using different clustering methods for layer 12 embeddings from dataset-fine-tuned BERT-base-cased models, while evaluating algorithm performance on identical data size.

of encoded concepts (number of words within the concept) per clustering algorithm in Figure 4. As depicted in the graph, Agglomerative clustering appears to have a propensity to generate a greater number of smaller clusters than K-Means which produces more medium-sized spherical clusters. Also, it could be noted that Agglomerative clustering resulted in a longer tail in the size distribution which might relate to its sensitivity to outliers.

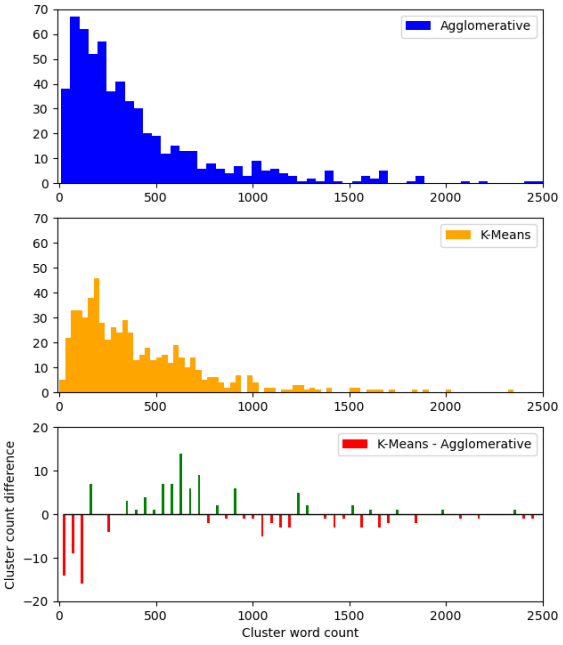


Figure 4: Histogram of cluster sizes for Agglomerative hierarchical clustering and K-Means on the same data. K-Means shows a heavier distribution (median 319 words per cluster), while Agglomerative clustering gave more small clusters (median 275) and a longer tail.

5.2 Concept discovery using scaled datasets

Given that K-Means results in higher quality concept discovery, we now ask the following question: *Does scaling the underlying dataset improve concept discovery?* To answer this, we proceed to compare the three clustering algorithms when the algorithm operates on as large of a dataset as possible within some external constraint. In our case, we used a maximum memory capacity of 500GB.

The results are presented in Table 2 for layers 10, 11 and 12 of fine-tuned BERT-base-cased models. The table illustrates the varying quantities of word representations that each clustering algorithm trains on. K-Means encountered no challenge in handling full datasets, whereas Agglomerative clustering necessitated sampling a subset of 220-250K words to operate within the memory confines. Regarding the Leaders variant of the Agglomerative clustering algorithm, an initial preprocessing of the dataset condensed the data into a reduced collection of 220-250K centroids which are then employed for clustering, as elaborated in Section 2.1. A binary search approach was used to determine the appropriate threshold value τ.

Our findings in Table 2 reveal a consistent superiority of the Leaders algorithm when compared to hierarchical Agglomerative clustering that operates in a data subset. This observation suggests that the initial pass of the Leaders algorithm through the data generates a representative dataset more apt for clustering, both in terms of coverage and alignment, and that data scaling can reveal better structure.

Possessing the capacity to handle the entire dataset without preprocessing, K-Means consistently outperformed the other two alternatives in terms of alignment and coverage across most scenarios. However, it does perform slightly poorly compared to Leaders in the case of CCG, specifically in layers 10 and 11. In CCG, words are tagged with complex linguistic categories that are composed hierarchically, reflecting the grammatical relationships and syntactic structures present in the text. Therefore it is plausible that a Leaders clustering potentially benefits from its ability to capture both overarching and nuanced relationships present in the hierarchical structure of the linguistic categories. Note that CCG coverage is also limited for this reason, stemming from the intricate and diverse range of syntactic functions that define these tags.

5.3 Computational complexity

Table 3 lists the average computational requirements of the three clustering algorithms in our experiments. The runtime is the total of the `user` and `sys` components of the output of Linux's `time` command. For peak memory usage, we employed

	Clustering	Size	Layer 10			Layer 11			Layer 12		
			Align. %	Cov. %	$\lambda_\theta(\mathcal{E},\mathcal{H})$	Align. %	Cov. %	$\lambda_\theta(\mathcal{E},\mathcal{H})$	Align. %	Cov. %	$\lambda_\theta(\mathcal{E},\mathcal{H})$
pos	**Agglomerative**	245K	44.2	55.6	0.50	46.2	60.0	0.53	47.8	60.0	0.54
	Leaders	906K	**59.2**	**64.4**	**0.62**	61.7	66.7	0.64	62.0	71.1	0.67
	K-Means	906K	58.8	**64.4**	**0.62**	**64.8**	**68.9**	**0.67**	**68.3**	**75.5**	**0.72**
ccg	**Agglomerative**	222K	63.3	19.1	0.41	65.8	21.0	0.43	67.2	22.6	0.45
	Leaders	923K	**79.0**	20.9	**0.50**	**78.3**	**25.1**	**0.52**	73.5	25.5	0.50
	K-Means	923K	75.2	**23.2**	0.49	75.3	22.8	0.49	**76.8**	**25.9**	**0.51**
sem	**Agglomerative**	223K	41.6	49.2	0.45	45.5	52.2	0.49	46.8	50.7	0.49
	Leaders	797K	50.5	59.7	0.55	52.8	58.2	0.56	58.2	61.2	0.60
	K-Means	797K	**57.6**	**62.7**	**0.60**	**61.2**	**59.7**	**0.60**	**68.0**	**67.2**	**0.68**

Table 2: Evaluating clustering methods across layers [10, 11, and 12] utilizing fine-tuned BERT-base-cased models, while operating within a memory constraint of 500GB. Align % = the percentage of encoded concepts that match the human-defined concepts within each task. Cov. % = percentage of distinct human-defined concepts that are acquired within the latent space. $\lambda_\theta(\mathcal{E},\mathcal{H})$ corresponds to an overall score combining the alignment percentage and concept coverage of human-defined concepts within a given task.

Python's `memory_profiler`.[5]

	Clustering	Size	**Runtime** (s)	**Memory** (GB)
pos	Agglomerative	245K	49,709	450.38
	Leaders	906K	50,967	421.43
	K-Means	906K	**32,461**	**13.59**
ccg	Agglomerative	223K	39,045	371.40
	Leaders	797K	60,599	443.94
	K-Means	797K	**37,930**	**16.17**
sem	Agglomerative	222K	35,401	375.76
	Leaders	923K	49,141	394.47
	K-Means	923K	**28,991**	**13.00**

Table 3: Runtime and memory requirements per clustering method and dataset, averaged across layers 10–12 for the results in Table 2

As the results clearly show, K-Means demonstrates superior time efficiency and remarkably low memory requirements compared to the other two alternatives, highlighting its potential for scalability. Please note that the numbers for the Leaders algorithm solely pertain to the two-stage clustering process and do not account for the binary search procedure required to determine the appropriate threshold τ to meet the memory limitation of 500GB.

5.4 Cross-architectural comparison

Do our findings generalize across models? We reproduced the BERT-base-cased experiments comparing the clustering approaches using the final layer of RoBERTa and XLM-RoBERTa. Despite the shared foundation of transformer-based pre-trained language models, these models vary in training regime, including data, optimization functions, pre-processing, and hyperparameters, among other factors. Our findings, shown in Table 4, revealed a certain trend: K-Means clustering consistently outperformed Agglomerative clustering across all scenarios. While the distinction between K-Means and the Leaders algorithm was less pronounced, K-Means remained the preferred choice due to computational requirements and potential for scalability.

[5] https://github.com/pythonprofilers/memory_profiler

	Clustering	Size	Layer 12		
			Align. %	Cov. %	$\lambda_\theta(\mathcal{E},\mathcal{H})$
bert	Agglomerative	245K	47.8	60.0	0.54
	Leaders	906K	62.0	71.1	0.67
	K-Means	906K	**68.3**	**75.5**	**0.72**
roberta	Agglomerative	245K	37.8	64.4	0.51
	Leaders	906K	51.7	64.4	0.58
	K-Means	906K	**56.8**	**77.8**	**0.67**
xlm-r	Agglomerative	245K	44.0	57.8	0.51
	Leaders	906K	**56.5**	**64.4**	**0.60**
	K-Means	906K	56.3	**64.4**	**0.60**

Table 4: Comparing aligned clusters and concepts using different clustering methods in BERT, RoBERTa and XLM-RoBERTa language models on POS task

When comparing different architectures, we observed that the concepts of BERT exhibit a stronger alignment with human-defined concepts in comparison to other models. For instance, when employing K-Means clustering, the percentage of concepts aligned in BERT is 68.3. We note that BERT shows a higher alignment of concepts across nearly all the tags compared to XLM-RoBERTa and RoBERTa, as shown in Figure 5. These findings suggest that concepts within BERT may display a greater level of redundancy compared to RoBERTa and XLM-RoBERTa. Our finding resonates with Durrani et al. (2020) who also found information to be more redundantly stored in BERT-base-cased model as opposed to RoBERTa and other PLMs.

6 Applications

Given the larger scale that K-Means clustering offers, we present preliminary results on potential

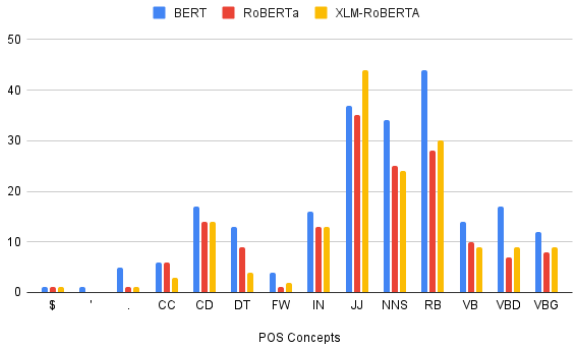


Figure 5: Number of aligned concepts for selected POS tag across different models. More results across various models and clusterings can be found in Appendix C.

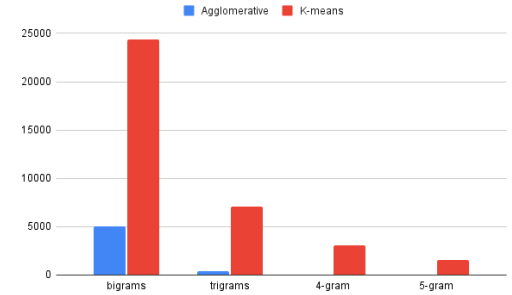


Figure 6: Number of phrasal units (2- to 5-grams) discovered using hierarchical Agglomerative clustering and K-means in the latent concepts of the BERT-SST model.

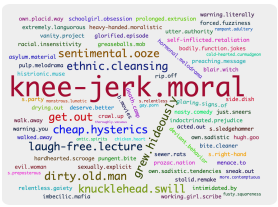
future directions in latent concept discovery that were previously not possible. Specifically, we look at concepts beyond the level of individual words, and also present results on two large language models, which have now become the foundation for many important leaps in NLP.

6.1 Phrasal-level interpretability

While it might suffice to focus solely on words when analyzing latent spaces for sequence labeling tasks, it becomes necessary to capture extensive contextual dependencies that span across longer ranges when dealing with sentence-level tasks, e.g. complex natural language understanding challenges like those found in the GLUE benchmark (Wang et al., 2018). Such tasks often revolve around extended word spans which play a crucial role in the task, referred to as "rationales" (DeYoung et al., 2020). For instance, in the context of sentiment classification for the following sentence, the highlighted spans specifically define a positive sentiment:

`In this movie, ... Plots to take over the world. The acting is great! The soundtrack is run-of-the-mill, but the action more than makes up for it.`

We demonstrate that our scaled-up concept discovery upgrades the framework of latent concept analysis to sentence labeling tasks such as Sentiment Classification. Figure 6 illustrates a comparison of the number of phrasal units (2 to 5-grams) found within the latent concepts of the BERT-SST model when using Agglomerative and K-Means clustering. The number of phrasal units discovered using K-Means improves significantly. Figure 7 shows example polarized concepts used in predicting negative (a) and positive (b) sentiment. We leave a detailed exploration of latent concept analysis for sentence-level tasks for the future.



(a) Negative Sentiment

(b) Positive Sentiment

Figure 7: Examples of encoded concepts in BERT-SST model. Tokens that are part of the same phrase are separated by periods.

6.2 Concept discovery for LLMs

Now we turn attention to latent concept discovery in large language models which our work facilitates. We investigate if `Llama-2-7B` and `Llama-2-7B-chat` exhibit any alignment to part-of-speech (POS) tags, similar to the models presented before. Figure 8 shows the alignment, coverage and 0.95-alignment score for the discovered concepts using the activations of the two LLMs. The chat-tuned version of Llama starts lower than its base sibling in alignment, but increases towards that later layers. Even though the difference is not very significant, we envisage that a suitable prompt that encourages the LLM to classify words into their respective POS tags would steer the alignment of the chat version higher. We leave this exploration of the role of the prompt for future work.

We also repeat the experiments that look at different underlying dataset sizes to verify that data scale is essential for latent concept discovery in LLMs. We recompute the alignment results for layer 20 of `Llama-2-7B` when we filter out any input word

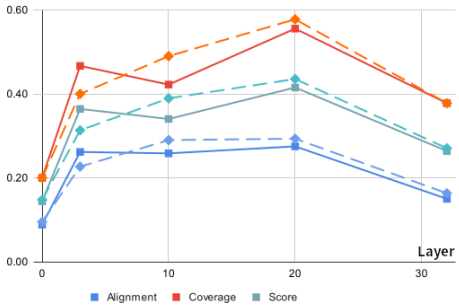


Figure 8: POS alignment, coverage and score for K-Means discovered concepts using layers 0, 3, 10, 20 and 32 in `Llama-2-7B` (solid) and `Llama-2-7B-chat` (dashed), showing a slightly better alignment in later layers in the chat-tuned variant.

that occurs less than 5 times or more than 1000 times (similar to the filtering done for Agglomerative clustering in Dalvi et al. (2022)). This results in a 50% reduction in the input data size. Table 5 shows a big drop in alignment and coverage using the reduced data, reinforcing that a lot of the lingual structure could be removed if data is pre-processed for size.

	Align.%	**Cov.%**	$\lambda_\theta(\mathcal{E}, \mathcal{H})$
full	**27.50**	**55.56**	**0.41**
filtered	22.17	28.89	0.25

Table 5: Alignment, coverage and 0.95-alignment score for layer 20 of `llama-2` when using the **full** data, and when the data is **filtered** based on word frequency. The results show the benefits of working on the full-scale of data for latent concept discovery.

7 Related Work

Clustering of representations from neural language models has been vital in a large number of studies to improve downstream NLP task and analyze language models. For instance, Aharoni and Goldberg (2020) cluster sentence embeddings to showcase the separation of domains in the embedding spaces of language models. Fei et al. (2022) use clustering to improve zero-shot text classification PLMs. Gupta et al. (2022) utilized deep clustering of representations to improve zero-shot performance on Part-of-Speech and Constituency label induction. Zhang et al. (2022); Thompson and Mimno (2020) cluster contextual embeddings to improve topic modeling of textual corpora. Michael et al. (2020); Dalvi et al. (2022); Fu and Lapata (2022) explored latent spaces of models to analyze the knowledge learned within a model. Sajjad et al. (2022b); Alam et al. (2023) compared them to the traditional and newly-discovered human ontologies. More recently Mousi et al. (2023) used LLMs to annotate latent spaces learned within pre-trained LMs. Our work addresses an important challenge underpinning these and similar works: the high computational cost of clustering in large embedding spaces.

8 Conclusion

Concluding this study, our exploration into uncovering latent concepts within the embedding space of pre-trained language models represents just the initial phase towards comprehending these models and establishing trust in their functionality. Our findings underscore the effectiveness of employing clustering of contextualized representations to unveil meaningful concepts that resonate with human comprehension. Furthermore, we highlight the viability of utilizing K-Means algorithm to handle expansive datasets, thereby facilitating the analysis of larger models and more intricate concepts.

Moving forward, there remains a compelling avenue for delving deeper into the interpretability of language models beyond the granularity of individual words. This involves unraveling the representation and utilization of complex linguistic constructs for inference purposes. Also, our exploration of the Leaders algorithm could potentially expand beyond our use with Agglomerative clustering. For instance, an intriguing avenue for future research includes enhancing the scalability of K-Means, pushing its limits to accommodate even more extensive datasets.

Limitations

The results presented in the paper mainly revolve around the proposed metric that combines alignment and coverage of human-defined ontologies. Although this metric serves as a reliable proxy for assessing the quality of clustering, it may not explicitly capture other dimensions of clustering that are equally important but not accounted for. Furthermore, we have limited our experimentation to three clustering algorithms among the many available, and it is conceivable that there are other algorithms that may result in even better quality at similar or lower computational cost than K-Means. Finally, our applications section mainly presents high level results without very deep exploration of the underlying hyperparameters, as these could be complete works in their own regard.

References

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The parallel meaning bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '17, pages 242–247, Valencia, Spain.

Charu C Aggarwal et al. 2015. *Data mining: the textbook*, volume 1. Springer.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Firoj Alam, Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Khan, Abdul Rafae, and Jia Xu. 2023. Conceptx: A framework for latent concept analysis. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI, Poster presentation)*, pages 16395–16397.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451. Association for Computational Linguistics.

Fahim Dalvi, Abdul Rafae Khan, Firoj Alam, Nadir Durrani, Jia Xu, and Hassan Sajjad. 2022. Discovering latent concepts learned in BERT. In *International Conference on Learning Representations*.

Fahim Dalvi, Avery Nortonsmith, Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, and James Glass. 2019. Neurox: A toolkit for analyzing individual neurons in neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):9851–9852.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '19, pages 4171–4186, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

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

Nadir Durrani, Fahim Dalvi, Hassan Sajjad, Yonatan Belinkov, and Preslav Nakov. 2019. One size does not fit all: Comparing NMT representations of different granularities. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, ACL '19, pages 1504–1516, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, and Fahim Dalvi. 2021. How transfer learning impacts linguistic knowledge in deep NLP models? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4947–4957, Online. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Firoj Alam. 2022. On the transformation of latent space in fine-tuned nlp models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1495–1516, Abu Dhabi, UAE. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. 2020. Analyzing individual neurons in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, EMNLP, pages 4865–4880, Online. Association for Computational Linguistics.

Yu Fei, Zhao Meng, Ping Nie, Roger Wattenhofer, and Mrinmaya Sachan. 2022. Beyond prompting: Making pre-trained language models better zero-shot learners by clustering representations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8560–8579, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yao Fu and Mirella Lapata. 2022. Latent topology induction for understanding contextualized representations.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

K Chidananda Gowda and G Krishna. 1978. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern recognition*, 10(2):105–112.

Vikram Gupta, Haoyue Shi, Kevin Gimpel, and Mrinmaya Sachan. 2022. Deep clustering of text representations for supervision-free probing of syntax. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10720–10728.

J. Hartigan. 1975. *Clustering Algorithms*. John Wiley and Sons, New York.

Julia Hockenmaier. 2006. Creating a CCGbank and a wide-coverage CCG lexicon for German. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, ACL '06, pages 505–512, Sydney, Australia. Association for Computational Linguistics.

Xin Jin and Jiawei Han. 2010. *K-Means Clustering*, pages 563–564. Springer US, Boston, MA.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. 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 4364–4373, Hong Kong, China. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL '19, pages 1073–1094, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

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

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Julian Michael, Jan A. Botha, and Ian Tenney. 2020. Asking without telling: Exploring latent ontologies in contextual representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, EMNLP '20, pages 6792–6812, Online. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the ICLR Workshop*, Scottsdale, AZ, USA.

Basel Mousi, Nadir Durrani, and Fahim Dalvi. 2023. Can LLMs facilitate interpretation of pre-trained language models? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3248–3268, Singapore. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Hassan Sajjad, Nadir Durrani, and Fahim Dalvi. 2022a. Neuron-level interpretation of deep NLP models: A survey. *Transactions of the Association for Computational Linguistics*, 10:1285–1303.

Hassan Sajjad, Nadir Durrani, Fahim Dalvi, Firoj Alam, Abdul Rafae Khan, and Jia Xu. 2022b. Analyzing encoded concepts in transformer language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '22, Seattle, Washington, USA. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. 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.

Laure Thompson and David Mimno. 2020. Topic modeling with contextualized word representation clusters.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. 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. Association for Computational Linguistics.

Zihan Zhang, Meng Fang, Ling Chen, and Mohammad Reza Namazi Rad. 2022. Is neural topic modelling better than clustering? an empirical study on clustering with contextual embeddings for topics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3886–3893, Seattle, United States. Association for Computational Linguistics.

Appendix

A Linguistic Concepts

We used parts-of-speech tags (48 concepts) using Penn Treebank data (Marcus et al., 1993), semantic tags (73 concepts) (Abzianidze et al., 2017), and CCG super tags (1272 concepts). Please see all the concepts below in Tables 6 and 7. This provides a good coverage of linguistic concepts including morphology, syntax and semantics.

B Sequence Tagger

We performed fine-tuning on the pre-trained language models for each of the three tasks (POS, CCG and SEM tagging) used in our analysis. This entails adjusting the latent space of the models towards the output classes, allowing us to evaluate clustering algorithms through the alignment function outlined in Section 3. We used standard splits for training, development and test data for the tasks. The splits to preprocess the data were released with Liu et al. (2019a) on github.[6] See Table 8 for statistics and classifier accuracy for BERT-base-cased model. Appendix A presents a comprehensive list of human-defined concepts within these ontologies.

C Comparing Architectures

We reproduced experiments comparing various clustering approaches using the final layer of different models. In particular, we examined the outcomes from BERT, RoBERTa, and XLM-RoBERTa. In Figure 9 we plot number of encoded concepts per POS tag for encoded concepts obtained via Agglomerative and K-Mean Clustering. Concepts in BERT are redundantly stored. The patterns hold consistently.

#	Tag	Description
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential there
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, comparative
9	JJS	Adjective, superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NNP	Proper noun, singular
15	NNPS	Proper noun, plural
16	PDT	Predeterminer
17	POS	Possessive ending
18	PRP	Personal pronoun
19	PRP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	to
26	UH	Interjection
27	VB	Verb, base form
28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun
36	WRB	Wh-adverb
37	#	Pound sign
38	$	Dollar sign
39	.	Sentence-final punctuation
40	,	Comma
41	:	Colon, semi-colon
42	(	Left bracket character
43	)	Right bracket character
44	"	Straight double quote
45	’	Left open single quote
46	"	Left open double quote
47	’	Right close single quote
48	"	Right close double quote

Table 6: Penn Treebank POS tags.

[6] https://github.com/nelson-liu/contextual-repr-analysis

ANA (anaphoric)		**MOD (modality)**	
PRO	anaphoric & deictic pronouns: he, she, I, him	NOT	negation: not, no, neither, without
DEF	definite: the, loIT, derDE	NEC	necessity: must, should, have to
HAS	possessive pronoun: my, her	POS	possibility: might, could, perhaps, alleged, can
REF	reflexive & reciprocal pron.: herself, each other	**DSC (discourse)**	
EMP	emphasizing pronouns: himself	SUB	subordinate relations: that, while, because
ACT (speech act)		COO	coordinate relations: so, {,}, {;}, and
GRE	greeting & parting: hi, bye	APP	appositional relations: {,}, which, {(}, —
ITJ	interjections, exclamations: alas, ah	BUT	contrast: but, yet
HES	hesitation: err	**NAM (named entity)**	
QUE	interrogative: who, which, ?	PER	person: Axl Rose, Sherlock Holmes
ATT (attribute)		GPE	geo-political entity: Paris, Japan
QUC	concrete quantity: two, six million, twice	GPO	geo-political origin: Parisian, French
QUV	vague quantity: millions, many, enough	GEO	geographical location: Alps, Nile
COL	colour: red, crimson, light blue, chestnut brown	ORG	organization: IKEA, EU
IST	intersective: open, vegetarian, quickly	ART	artifact: iOS 7
SST	subsective: skillful surgeon, tall kid	HAP	happening: Eurovision 2017
PRI	privative: former, fake	UOM	unit of measurement: meter, $, %, degree Celsius
DEG	degree: 2 meters tall, 20 years old	CTC	contact information: 112, info@mail.com
INT	intensifier: very, much, too, rather	URL	URL: http://pmb.let.rug.nl
REL	relation: in, on, 's, of, after	LIT	literal use of names: his name is John
SCO	score: 3-0, grade A	NTH	other names: table 1a, equation (1)
COM (comparative)		**EVE (events)**	
EQU	equative: as tall as John, whales are mammals	EXS	untensed simple: to walk, is eaten, destruction
MOR	comparative positive: better, more	ENS	present simple: we walk, he walks
LES	comparative negative: less, worse	EPS	past simple: ate, went
TOP	superlative positive: most, mostly	EXG	untensed progressive: is running
BOT	superlative negative: worst, least	EXT	untensed perfect: has eaten
ORD	ordinal: 1st, 3rd, third	**TNS (tense & aspect)**	
UNE (unnamed entity)		NOW	present tense: is skiing, do ski, has skied, now
CON	concept: dog, person	PST	past tense: was baked, had gone, did go
ROL	role: student, brother, prof., victim	FUT	future tense: will, shall
GRP	group: John {,} Mary and Sam gathered, a group of people	PRG	progressive: has been being treated, aan hetNL
DXS (deixis)		PFT	perfect: has been going/done
DXP	place deixis: here, this, above	**TIM (temporal entity)**	
DXT	temporal deixis: just, later, tomorrow	DAT	full date: 27.04.2017, 27/04/17
DXD	discourse deixis: latter, former, above	DOM	day of month: 27th December
LOG (logical)		YOC	year of century: 2017
ALT	alternative & repetitions: another, different, again	DOW	day of week: Thursday
XCL	exclusive: only, just	MOY	month of year: April
NIL	empty semantics: {.}, to, of	DEC	decade: 80s, 1990s
DIS	disjunction & exist. quantif.: a, some, any, or	CLO	clocktime: 8:45 pm, 10 o'clock, noon
IMP	implication: if, when, unless		
AND	conjunction & univ. quantif.: every, and, who, any		

Table 7: Semantic tags.

Task	Train	Dev	Test	Tags	F1
POS	36.5K	1802	1963	48	96.81
CCG	39.1K	1908	2404	1272	95.24
SEM	36.9K	5301	10600	73	96.32

Table 8: Statistics of the datasets used in the experiments, the number of concepts (tags) for each task and the performance of the fine-tuned models

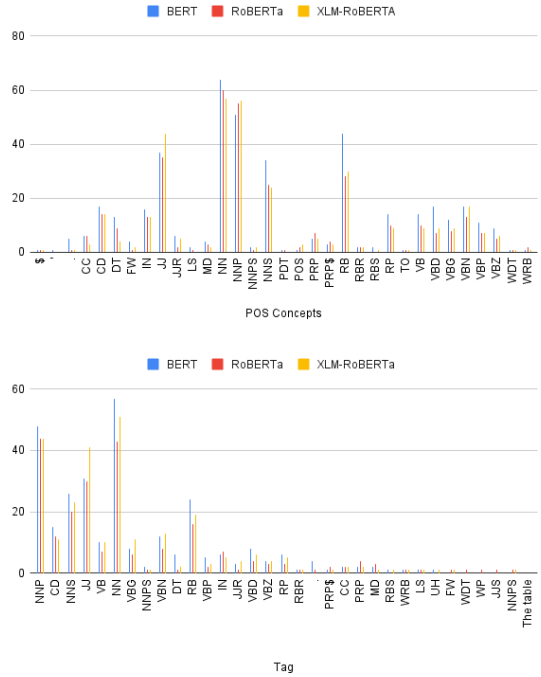


Figure 9: Number of aligned concepts per POS tag across different models