Nested State Clouds: Distilling Knowledge Graphs from Contextual Embeddings

Bachelor's Project Thesis

Paul Bricman, s3908194, p.a.bricman@student.rug.nl,
Supervisors: Prof. Dr. Herbert Jaeger

Abstract: Interpretability techniques help ensure the safe deployment of ML models into production by providing practitioners with diverse debugging tools, yet the inner workings of large models remain elusive. In this work, we propose a novel interpretability technique which can be used to distill sparse knowledge graphs from a model's high-dimensional embeddings. This technique, termed Nested State Clouds (NSC), takes advantage of the relative spatial layouts of state clouds in latent space (e.g. "fruit" contextual embeddings appear to engulf "apple" ones). We successfully apply NSC to BERT, and recover an ontology of concepts grounded in the model's latent space.

1 Introduction

In the past years, ML models have been claimed to reach human parity on a range of tasks which were deemed challenging only years prior. For instance, machine translation is deemed on par with human translators on popular language pairs, scientific ML models are more accurate than explicit hand-crafted ones in a growing range of fields, while RL agents have outperformed professionals in multiple video games and industrial control applications.

The resounding success of recent ML work has in large part been attributed to the newly-gained ability of ML models to automatically extract relevant features from input data, rather than making use of hand-crafted features. This has proven an instrumental goal in advancing the state-of-the-art on the vast majority of ML tasks. Concretely, the automatically-derived features are represented through the specific activation patterns of hidden layers as information propagates through the ML model. Besides early layers being able to already extract surface input features, a recurring finding has been the fact that input representations become increasingly abstract with each sequential layer, before collapsing again to the concrete particularities of the output towards the final layers. When such representations are continuous and dense, they are referred to as embeddings.

In the context of our increased reliance on ML models on a societal level, the field of explainable AI (XAI) investigates methods for interpreting the inner workings of such models, which otherwise lack clear internal rules due to being largely trained on raw data. Techniques of this kind help researchers debug such ML systems, ensuring their safe use in practice. A pervasive trade-off in XAI, however, is the conflict between formulating explanations which (1) accurately reflect the actual processing performed by ML models (i.e. functionally-grounded), yet (2) are highly comprehensible and intelligible for humans (i.e. human-grounded). This echoes a constant conflict faced in machine translation, where models should (1) adequately preserve input meaning (i.e. adequacy), while (2) producing a coherent translation in itself (i.e. fluency). Framing explainable AI as a translation task from machine to human representations has proven a useful lens for understanding the obstacles faced by the current work.

Given the central role of input representations in recent ML models, a large body of advances in XAI has focused on distilling high-dimensional embeddings into a form which is cognitively ergonomic for humans. As the embeddings themselves arguably lack meaning when separated from input or output data, the vast majority of such work has specifically attempted to highlight how a certain ML model relates inputs and outputs by means of embeddings.

For instance, prior work has highlighted toxic gender biases in word embeddings (e.g. "woman" being represented as closer in meaning to "nurse" than "programmer" by an ML model), which promptly led to debiasing techniques being developed by the NLP community. Moreover, methods have been developed to construct ontologies from embeddings by means of hierarchical clustering, in order to better understand how the underlying ML model groups concepts together. As another line of work, behavioral and structural probes have been employed to locate the "site" of various computations (e.g. part-of-speech tagging in NLP), by means of relating embeddings from different layers to cruder external representations (e.g. part-of-speech). Additionally, methods have been suggested to explicitly represent part-whole relations using embedding "columns", hinting at future ML models which are partially explainable themselves, even before making use of post-hoc XAI tools.

However, even when interpretability techniques focus on directly relating inputs to outputs, embeddings are often involved as mediators. For instance, input feature explanations highlight which particular aspects of the input data have been most influential in yielding the output. Alternatively, techniques based on counterfactuals and adversarial examples aim to find marginally different inputs which cause massive changes in output. As a particularly ergonomic family of explanations, methods have also been developed to incentivize models to directly explain themselves in natural language. Finally, another effective technique is based on extracting knowledge graphs by inferring entity-relationship-entity triples directly via (masked) language modeling.

In this context, we extend the existing toolkit of interpretability techniques with a novel approach called Nested State Clouds (NSC). This technique can be used to distill highly-comprehensible knowledge graphs directly from sets of high-dimensional embeddings, with no modality constraints. In other words, given an ML model and an auxiliary dataset, NSC can be used to automatically organize concepts in a part-whole hierarchy which, in large part, reflects the model's internalized knowledge. As investigated in this paper, this novel interpretability technique can be used to surface learned ontologies from sequence-to-sequence models (e.g. BERT, GPT). However, NSC can also be applied to arbitrary classification models (e.g. ViT), by creating state clouds from individual class member embeddings. Given this, an important benefit of NSC is its modality-agnostic nature, in stark contrast to methods which only focus on distilling knowledge graphs from text. By operating with spatial layouts of embeddings, NSC can distill high-dimensional representations which have been abstracted away from particular modalities.

NSC works by first generating a state cloud of contextual embeddings for each given entity, representing the different underlying meanings of the symbol in different contexts. For instance, the concept "fruit" can refer to a host of different objects, depending on the context of use – variation which is captured by distinct contextual embeddings. Next, NSC compactly represents the overarching shape of the resulting state cloud as a high-dimensional ellipsoid, instead of a set of embeddings, which often results in orders-of-magnitude lower memory footprint. While the ellipsoids resemble PCA and SVD outputs, we opted for using conceptors as compact high-dimensional objects due to existing literature investigating meaningful ways of relating them to each other. Specifically, it has been posited – yet never before tested empirically, to the best of our knowledge – that conceptors benefit from an inherent abstraction ordering, a means of comparing two such objects in terms of their level of abstraction. Loosely speaking, a conceptor which spatially engulfs another can be said to be more abstract, as it encompasses a broader region of space than the other one.

After generating state clouds, representing them as conceptor objects, and conducting pairwise comparisons of abstraction, a search algorithm is employed to find a directed graph which accurately represents the estimated relations of abstraction. In contrast to simply constructing a directed graph by adding a new arc for each positive abstraction relation, a search algorithm allows us to better deal with noise. Additionally, the search framing enables us to specify additional "nice-to-have" properties of the desired output graph. For instance, we penalize high numbers of arcs, parents per node, and children per node, in an attempt to keep the output explanations sparse and highly intelligible. In this, the objective function of the search algorithm essentially provides a "slider" between functionally-grounded and human-grounded explanations.

Attempting to place NSC in the existing landscape of interpretability techniques, we note that our approach can generate global explanations (i.e. holistically describing the model's processing across inferences) which are provided post-hoc (i.e. after training the model). This is in contrast to those XAI techniques which yield local explanations (i.e. describing the way a particular inference unfolds across layers) and those which are provided during the actual training of inherently interpretable models. Beyond this general placement of NSC in the XAI literature, we point out relevant similarities and differences to prior art throughout the paper.

Our contributions are as follows:

- We provide qualitative evidence highlighting the connection between the spatial layout of nested state clouds and the abstraction relation of the concepts they represent;
- We formulate an novel algorithm for flexibly distilling a set of high-dimensional state clouds into a compact directed graph which depicts part-whole relations;
- We draw evidence-based observations on the way individual symbols relate to concepts.

2 Methods

2.1 Model

As the object of our interpretation technique, we chose a pretrained BERT model, short for Bidirectional Encoder Representations from Transformers. For completeness, BERT is a transformer model which maps a set of subword tokens to another set of such tokens. BERT has been originally trained on two different natural language processing objectives. First, it has been tasked with a masked language modeling (MLM) objective. This refers to the task of reconstructing a short input text which has been intentionally corrupted. The corruption typically consists in eliminating (i.e. masking) a random proportion of the tokens contained in the input text (e.g. "BERT is a transformer model." might be corrupted as "BERT is a [MASK] model."). Given this, the MLM task consists in reconstructing the pre-corruption text from the corrupted version.

The second objective employed in training BERT is a next sentence prediction (NSP) task. Given a pair of two sentences, BERT is tasked with predicting whether they are consecutive in the original text. The combination of those two conceptually simple objectives has been shown to help BERT learn rich semantic representations of the text being processed, as an instrumental goal in solving the two tasks. For instance, mean-pooling token embeddings across texts has been shown to be highly effective in downstream information retrieval tasks based on vector similarity. Moreover, mean-pooling token embeddings of a text and comparing the result with the mean-pooled embeddings of a set of labels (e.g. science, politics, economics), has been shown to be a competitive baseline in text classification. Alternatively, BERT models fine-tuned on limited data from other tasks (e.g. natural language inference) had yielded state-of-the-art performance in multiple tasks.

Internally, BERT represents each token as an embedding of dimensionality 768. As the model consists of a repetitive sequence of layers, the set of embeddings which represents tokens is adjusted from one layer to the next using multi-head attention mechanisms. It is precisely those token embeddings which we are trying to distill knowledge graphs from using the NSC approach. Specifically, as often done in prior art, we are attempting to interpret the set of embeddings which are generated in the *last* BERT layer. This has been hypothesized to contain high-level features extracted the input tokens which are based in large part on the tokens' contexts, after extensive processing in the earlier layers of the model.

We opted for BERT due to its widespread use in industry applications and its large number of derivative models (e.g. RoBERTa, ALBERT, distilBERT, etc.). BERT takes in a sequence of subword tokens as input and reconstructs it as output, while generating a unique contextual embedding for each token. Crucially, the same token can be attributed different embeddings in different contexts (e.g. "she" referring to different people). In practice, the contextual embeddings of individual tokens are mean-pooled together to yield an overarching document embedding. However, here we focus only on the contextual embeddings of individual tokens or at most short sequences of them which form a noun phrase (e.g. "orange juice").

2.2 Data

As NSC requires an auxiliary dataset for generating state clouds of contextual embeddings, we employ one of the datasets which have been used for training the BERT model, namely BookCorpus. This dataset consists of a variety of public domain books across different genres, and provides many different contexts for tokens to appear in.

We focus our investigation on relating a set of 100 hand-picked concepts to each other. For each concept, we extract all contexts in which they appear verbatim in the dataset. A context is defined as the span of text starting 300 characters before and ending 300 characters after the concept occurence. Additionally, we trim the incomplete beginning and ending sentences (i.e. trailing) from each context, leaving in only complete sentences surrounding the concept occurence.

For each context, we extract the contextual embedding of the concept occurence, obtaining a set of such embeddings for each concept. The cardinality of each set depends on the frequency of occurence of the concept in the dataset. We further filter our set of concepts based on the size of the set of contextual embeddings, eliminating concepts which had fewer occurences than the number of BERT embedding dimensions (i.e. 768). We cover difficulties in handling sparser state clouds in the discussion.

2.3 Conceptors

From each remaining state cloud representing the set of concept nuances used in the dataset, we learn a conceptor. For completeness, a conceptor is a mathematical object which models the distribution of state cloud in its space. However, conceptors do *not* represent the *density* of embeddings in space using a probability density function. Rather, a conceptor represents the *dimensions* across which the state cloud spreads most across space, together with the spread associated with each dimension. This information can be compactly represented in a square matrix whose dimensionality matches the one of the space populated by the state cloud.

Both conceptors and Principal Component Analysis (PCA) make use of the correlation matrix of the state cloud. However, an additional parameter appears in the case of conceptors. Specifically, a conceptor also requires an aperture to be defined. The aperture α is a parameter which dictates the extent to which the state cloud is reflected in the associated conceptor object. For increasing aperture values, the conceptor matrix approaches the identity matrix. For decreasing aperture values, the conceptor matrix approaches the zero matrix.

Obtaining a conceptor from a state cloud is straight-forward and computationally cheap. Given the correlation matrix of the state cloud and a real value specified for the aperture parameter, the conceptor matrix can be obtained through the following closed-form equation:

$$C(R, \alpha) = R(R + \alpha^{-2} I)^{-1}.$$

In the present work, we only obtain conceptors from state clouds composed of contextual embeddings generated by the pretrained BERT model. Each state cloud contains contextual embeddings associated with one symbol composed of one or several tokens (e.g. "orange juice"), while each individual contextual embedding is associated with a specific occurence of the symbol in the text corpus. Given that we obtain conceptors from state clouds of BERT contextual embeddings, the dimensionality of the state cloud will match that of the embeddings, namely

$$d_{BERT} = 768.$$

We note that the interpretability technique we introduce does not require this specific

dimensionality. Rather, the specific value of d_{BERT} is only an artifact of the investigated model's own architecture. It is conceivable that NSC could be applied with minimal modifications to state clouds of lower or higher dimensionality. In fact, some of the experiments used to introduce conceptors as a mathematical object make use of state clouds of only several dimensions.

Additionally, it is useful to note the large difference in terms of memory footprint observed between a state cloud of BERT embeddings and a conceptor matrix derived from it. A state cloud containing $n_embs = 10^6$ BERT embeddings of dimensionality $d_{BERT} = 768$ naively requires $n_{embs} \cdot d_{BERT} = 7.68 * 10^8$ floating point values to fully represent. In contrast, the conceptor matrix obtained from the same state cloud, given a certain aperture, is a square matrix with d_{BERT} rows and columns. Hence, it only needs $d_{BERT}^2 \approx 5.89 * 10^5$ floating point values to be represented. In this specific case, the conceptor represents the state cloud with an approximately 10^3 times smaller memory footprint. Moreover, the memory footprint of the conceptor matrix is constant with respect to the cardinality of the state cloud. It is only the accuracy of representing the state cloud which increases with more samples in the form of new contextual embeddings, as the correlation matrix converges.The implication of this is that state clouds associated with symbols which have relatively high frequency in the auxiliary dataset get represented through conceptor matrices of the same size as those associated with symbols with relatively low frequency.

2.4 Abstraction Ordering

For each pair of conceptors (C_1, C_2) obtained from different state clouds of contextual embeddings, we attempt to estimate how they relate to each other in terms of abstraction. We aim for determining whether (1) C_1 represents a concept which is more abstract than the one represented by C_2 (e.g. "fruit" $>$ "apple"), whether (2) it is the other way around (e.g. "apple" $<$ "fruit"), or whether (3) the two concepts represented lack a meaningful abstraction ordering (e.g. "fruit" $<>$ "galaxy"). Ideally, we would only want the final knowledge graph generated by NSC to represent relations of decreasing abstraction by means of arcs (e.g. "fruit" $>$ "apple"). This would lead to a readable knowledge graph which approaches a hierarchical structure.

In its original formulation, abstraction ordering of conceptors has two important characteristics. First, prior art only describes the three mutually-exclusive cases above, with hard limits. C_1 is described to be more abstract than C_2 if and only if the difference matrix $C_1 - C_2$ is positive definite. Due to the symmetry of abstraction ordering, $C_2 > C_1$ if and only if the difference matrix $C_2 - C_1$ is positive definite. Unfortunately, real data is noisy, making it extremely unlikely that the unambigious criterion of positive definiteness ever holds for conceptors obtained from non-synthetic data (e.g. BERT contextual embeddings). Besides the inevitable aleatoric noise associated with non-synthetic data, abstraction ordering in its original formulation is also hindered by the cumulative error introduced by limited machine precision when dealing with floating point values. Approaches from numerical methods, however, might help mitigate the impact of this second source of noise.

Besides requiring standards of precision and signal-to-noise ratio which are non-trivial to attain in practice, abstraction ordering in its original formulation also happens to provide hard cut-offs. C_1 can be determined to be more abstract than C_2, less abstract, or equally abstract. In practice, a continuous signal representing *how much* more abstract C_1 is compared to C_2 appears to be quite useful relative to the original ternary signal. In attempting to make use of an abstraction ordering signal which (1) is decently robust against the noise of real-world data, and (2) can be used to gauge the precise *magnitude* of the abstraction difference, we introduce a heuristic. The design of this heuristic has been informed by the fact that symmetric positive definite ma-

trices only have positive eigenvalues. This property has led to the idea of mean-pooling eigenvalues and using both the polarity and magnitude of the result as a proxy for abstraction ordering of two given conceptors.

Concretely, we estimate the abstraction ordering of (C_1, C_2) by means of the following heuristic:

$$f(C_1, C_2) = \frac{1}{d_{BERT}} \sum_{i=1}^{d_{BERT}} \lambda_i(C_1 - C_2).$$

To unpack, we first substract one conceptor matrix from the other. Second, we compute the mean of the eigenvalues the difference matrix. Intuitively, all eigenvalues of the difference matrix are positive if the first conceptor spatially engulfs the other, having higher spread than the second across all dimensions. In the context of this project, across all $d_{BERT} = 768$ dimensions. Conversely, all such eigenvalues are negative if the first conceptor is completely contained by the second across all dimensions. Inevitably, however, the two conceptors will exhibit one such relation across *some* dimensions, while simultaneously exhibiting the opposite in other dimensions. Hence, we average the eigenvalues in an attempt to reach a "consensus opinion" as to how the two conceptors are related to each other.

2.5 Graph Optimization

Given the pairwise estimates of abstraction ordering computed before, we conduct a graph optimization process. All candidate graphs considered are directed ones, while nodes are identified with concepts, and arcs indicate meronymous relations of abstraction (i.e. IS_A).

We attempt to solve the graph optimization task through the local search algorithm of simulated annealing (see 2.2). **Each candidate graph is represented through a Boolean adjacency matrix A. Specifically, A^k denotes the adjacency matrix of the candidate graph considered at step k of the graph optimization process. A^k_{ij} is a Boolean value indicating whether concept i links to concept j in the associated candidate graph of step k.** As an initial candidate graph, the optimization process starts with a fully-disconnected graph, where no concepts are related to each other. This is represented through an adjacency matrix full of null values, $A^0 = 0$. Then, we randomly sample a new graph proposal by randomly mutating the current graph – removing a previous arc or adding a new one. The acceptance probability is informed by a temperature schedule which linearly decreases from one to zero over the course of the search process, encouraging heavy exploration in the first epochs while using an increasingly conservative strategy towards the end.

The objective function which the search algorithm attempts to maximize is a linear combination of four different terms. Each term is a function of either or both (1) the adjacency matrix A^i which is identified with the state of the graph optimization process in step i, and (2) the matrix D containing pairwise estimates of abstract ordering. D_{ij} denotes the numerical estimate of abstraction between conceptor i and j. In other words,

$$D_{ij} = f(C_i, C_j)$$

. We note that the particular way D_{ij} is related to D_{ji} is determined by the choice numerical heuristic employed for abstraction ordering. In our case (i.e. mean of eigenvalues of difference matrix), $D_{ij} = -D_{ji}$, yet this is not necessarily the case when opting for other heuristics, as explored in the discussion (e.g. positive-negative eigenvalues ratio). Besides the two matrices just described which influence the objective function through the four terms whose description follows, the objective function is also influenced by the four coefficients which are used to weigh the four terms.

The first term is a function of both A^i and D. It is equal to the mean of the abstraction ordering estimates represented in the candidate graph by means of arcs. From now on, we refer to this term as *expressed abstraction* (EA). In case of a fully-disconnected graph (i.e. one in which no arc exists in the graph at all) represented by $A^i = 0$, $EA(A^i, D) = 0$. In contrast, in case of a fully-connected graph (i.e. one in which there exists an arc between any two nodes) repre-

sented by $A^i = 1$,

$$EA(A^i, D) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} D_{ij}.$$

In general, $min(D) \leq EA(A^i, D) \leq max(D)$. This is the only among the four terms of the linear combination which indicates functional-groundedness, as it reflects the proportion of the abstraction identified in high-dimensional space which gets represented in the output knowledge graph.

The second term is only a function of A^i. It is equal to the proportion of arcs contained by the candidate graph represented by adjacency matrix A^i, relative to the maximum number of possible arcs n^2:

$$AD(A^i) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}.$$

We refer to this term as *arc density* (AD). In case of a fully-disconnected graph with $A^i = 0$, $AD(A^i) = 0$. In case of a fully-connected graph with $A^i = 1$, $AD(A^i) = 1$. In all other cases, $0 < AD(A^i) < 1$.

The third term is also only a function of A^i. It is equal to the mean difference between a node's children count (i.e. number of nodes connected via outbound arcs) and a target children count set in advance:

$$CE(A^i) = \frac{1}{n} \sum_{i=1}^{n} \left| (\sum_{j=1}^{n} A_{ij}) - target\_children \right|.$$

We refer to this term as *children error* (CE). In case of a graph with adjacency matrix A^i in which every node only has children equal in count to $target\_children$, $CE(A^i) = 0$. For all other graphs, $CE(A^i) > 0$.

The fourth and final term of the objective function is extremely similar to the previous one, yet it addresses the distribution of parent counts, rather than children counts. Concretely, it is equal to the mean difference between a node's parent count (i.e. number of nodes connected via inbound arcs) and a target parent count set in advance:

$$PE(A^i) = \frac{1}{n} \sum_{j=1}^{n} \left| (\sum_{i=1}^{n} A_{ij}) - target\_parents \right|.$$

Algorithm 2.1 Graph Search in NSC

```
$s \Leftarrow 0$ (fully-disconnected graph)
for $k = 0$ to $epochs$ do
    $T \Leftarrow 1 - \frac{k}{epochs}$
    $s_{new} \Leftarrow neighbor(s)$
    if $P(score(D, s), score(D, s_{new}), T) \geq random(0, 1)$ then
        $s \Leftarrow s_{new}$
    end if
end for
```

We refer to this term as *parent error* (PE). In case of a graph with adjacency matrix A^i in which every node only has parents equal in count to $target\_parents$, $PE(A^i) = 0$. For all other graphs, $PE(A^i) > 0$.

Besides the four terms which are functions of either or both A^i and D, the objective function also contain four coefficients meant to influence the relative weight of each term. We denote these as α, β, γ, and δ in order for the four terms. Given the four terms, the two additional targets for child and parent count, and the four weighing coefficients included in the linear combination, the objective function can finally be defined as:

$$\begin{aligned}
&score(A^i, D, target\_children, target\_parents) = \\
&\alpha EA(A^i, D) \\
&- \beta AD(A^i) \\
&- \gamma CE(A^i, target\_children) \\
&- \delta PE(A^i, target\_parents).
\end{aligned}$$

The result of the graph search is the final output of NSC: a graph which indicates how the underlying ML model relates concepts by means of contextual embeddings.

In the context of the present work, we have manually specified values for the four coefficients of the linear combination which comprises the graph optimization objective, α, β, γ, and δ. Besides those, we have also manually specified values for the two hyperparameters related to local graph structure, $target\_children$ and $target\_parents$. However, a more robust approach to specifying the values of those six hyperparameters would be

Algorithm 2.2 Nested State Clouds

```
for $s$ in $symbols$ do
    $a \Leftarrow contexts(s)$
    $b \Leftarrow cloud(a)$
    $c \Leftarrow conceptor(b)$
end for
for $i, c_i$ in $conceptors$ do
    for $j, c_j$ in $conceptors$ do
        $D_{ij} \Leftarrow f(c_i, c_j)$
    end for
end for
$s_{output} \Leftarrow graph\_search(D)$
```

to conduct a hyperparameter search. Concretely, one might resort to searching for appropriate values for those six hyperparameters – in addition to the number of graph optimization epochs, the temperature schedule, and the conceptor aperture – which successfully recover some presupposed relations of abstraction between the concepts being related by means of the resulting knowledge graph. We leave that for future work and expand on the possibility in the discussion.

3 Results

In this paper, we designed a new interpretability technique which can be used to extract knowledge graphs from state clouds of contextual embeddings. We have noticed that NSC is able to successfully recover commonsense relations of abstraction from raw text data (e.g. "apple" IS_A "fruit", "orange juice" IS_A "juice", see Fig. 3.1). Additionally, we have found that for a limited number of concepts to relate, the graph search is robust with respect to the starting state. Moreover, the legibility terms included in the linear combination which comprise the search objective (e.g. arc count) successfully nudge the search towards relatively sparse outputs. Finally, the graph search history profile exhibits proper foraging behavior, with fast increases in solution quality in the beginning, followed by a more conservative strategy which ends in marginal improvements towards the move to heavy exploitation (see Fig. 3.2).

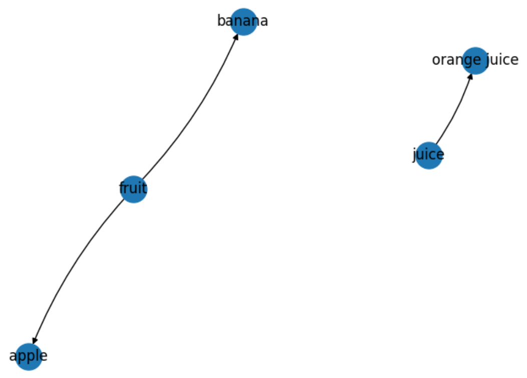

Figure 3.1: NSC output graph when applied to BERT using several symbols.

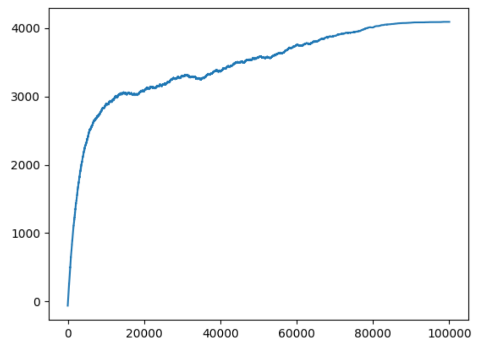

Figure 3.2: Candidate score by epoch during the local graph search.

4 Discussion

4.1 Potential issues

4.1.1 The same symbols can represent different concepts.

Upon inspecting low-dimensional state cloud projections, we observed the presence of distinctive clusters across latent space (see Fig. 4.1). For instance, the state cloud of the symbol "plant" appears to be populated by at least three clusters. To investigate this, we ran a K-means clustering ($K = 3$) procedure on the "plant" state cloud and surfaced the contexts which yielded contextual embeddings closest to the cluster centroids. Upon inspection of those contexts, we noticed that the context sets contained distinctive word senses, roughly corresponding to (1) plant objects, (2) the action of planting, and (3) factories (see Table 4.1).

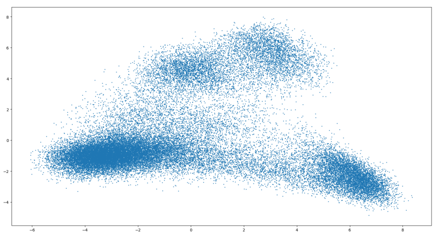

Figure 4.1: 2D PCA projection of the state cloud associated with the symbol "plant"

This diversity of meanings assumed by the same symbol across the text corpus casts doubt on our assumption of there being a one-to-one correspondence between symbols and graph nodes. An intermediate clustering step might be effective in decoupling different word senses and producing different state clouds, though the issue of how many senses are there per symbol is non-trivial. Similar to how words themselves appear to discretely quantize the otherwise continuous semantic space, finite word senses as "subsymbols" run into similar trade-offs between sparsity and accuracy.

Table 4.1: Context samples by K-means cluster of "plant" state cloud.

Cluster	Sample context
1	absently, i raised the blinds so that the plant was able to soak in the impromptu sunshine.
	i've brought you over a few macramé plant hangers to decorate your room.
2	i wanted to plant them myself.
	she'll just plant new ones and start all over again.
3	the computers running the plant were all infected, of course.
	it was plant shutdown for two weeks.

4.1.2 State clouds are non-linear.

While we employ conceptors as compact elliptical objects which approximate high-dimensional state clouds of contextual embeddings, their limited expressivity might fail to capture the intricate non-linear layout of real-world embeddings. Low-dimensional PCA projections of several state clouds of BERT embeddings radically diverge from Gaussian distributions, bringing into question the suitability of elliptical conceptors to represent them (see Fig. 4.2). However, we note that state clouds of less ambiguous terms (i.e. limited number of word senses) appear more well-formed. Non-linearities might arise mainly from diverse word senses being assumed by the same symbols.

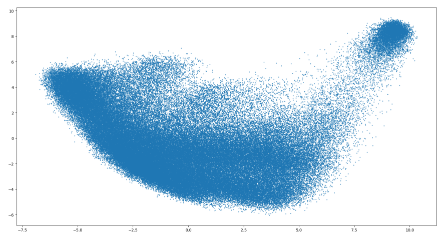

Figure 4.2: 2D PCA projection of the state cloud associated with the symbol "earth"

4.1.3 NSC requires many exemplars.

The central role of state clouds in NSC means that the technique is highly dependent on a large number of occurences and contexts for each concept analyzed. This makes it difficult to interpret the model's internal representations with respect to obscure tokens, as those are extremely rare in natural datasets. However, synthetic datasets might address this issue, provided the ability to synthesize a wide range of unique contexts for an arbitrary concept.

4.1.4 NSC is influenced by the choice of auxiliary data.

NSC does not only require the model under investigation in order to work. There is also the requirement of an auxiliary dataset which contains a wide range of exemplars

in different contexts. NSC then uses those exemplars in order to approximate the overarching state cloud with a conceptor object. However, the particular choice of dataset in which the concept instances are to be found has a high influence on the output knowledge graph.

For instance, consider two senses of the symbol "property." It might refer to (1) a characteristic, or (2) a real estate asset, among other senses. Different datasets might contain quantitatively distinct distributions of the meaning of "property" across latent space. One predominantly containing text about real estate matters might be skewed towards considering sense (2) above as the most pervasive one. In contrast, a dataset containing text about the physical properties of certain materials might be skewed in the other direction, towards considering sense (1) above as the most typical one.

One might be tempted to simply aim for large datasets in order to alleviate this concern. Unfortunately, the issue as hand is not that certain senses are not represented enough through exemplars in order for their meaning to be captured. Rather, it is specifically an issue of representativity in the resulting population of meanings. This has two implications, one concerning industry applications, and one concerning epistemics. The first is that, in practice, one would have to identify an auxiliary dataset which is representative enough of the concepts desired to be placed in the resulting knowledge graph. For instance, if one aims to investigate an ML model's internalized representations related to real estate matters, one concerning material science would be a poor choice of auxiliary dataset. The second implication hints at the fact that transparency tools like the ones used broadly in XAI can only meaningfully relate latent representations to the world models of people who authored the datasets. If one was to apply NSC on a book corpus, as we did, it would be unlikely for terms like "entity," "object", or "thing" to be organized in an ontology in the same way a logician might organize them.

4.1.5 NSC output graph is heavily influenced by the graph search objective.

The graph search process employed to output a knowledge graph is highly sensitive to the search objective. Reaching a balance between the functional-grounded terms (e.g. faithfulness to detected abstraction ordering) and human-grounded terms (e.g. sparsity) is difficult to achieve manually. Often, one component of the linear combination tends dominates the others (see Fig. 4.3 and Fig. 4.4). This balance is especially difficult to find with higher number of concepts, bringing the scalability of NSC into question. However, normalizing the objective's components based on the number of concepts being analyzed greatly improved robustness.

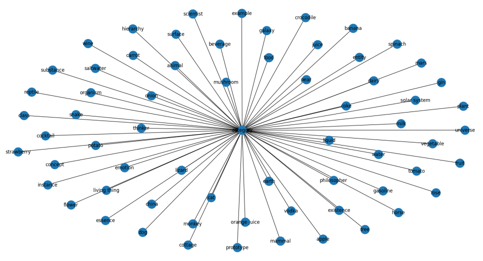

Figure 4.3: NSC output graph after undervaluing children count per node constraints.

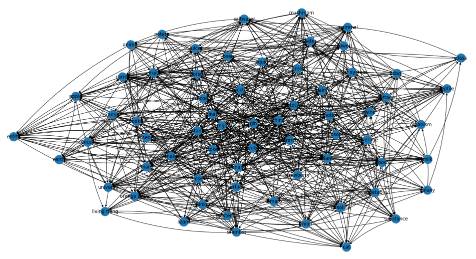

Figure 4.4: NSC output graph after undervaluing arc pruning.

4.2 Future work

4.2.1 Improved noise robustness

Conceptors obtained from state clouds of contextual embeddings are challenged by

several sources of noise. For one, the contextual embeddings themselves are noisy, as they have been generated by the pretrained BERT model, which in turn has been trained on a noisy real-world corpus comprised of books. In addition, the number of exemplars available for a given concept in the auxiliary corpus is implicitly limited, only depicting a partial representation of the underlying sampled distribution of meanings. Moreover, the closed-form step of obtaining conceptors from given state clouds is itself vulnerable to limited machine precision, questioning the suitability of the resulting conceptor object in representing the state cloud.

To this end, we argue that improved noise robustness would be a worthwhile future step to consider in improving the utility of NSC as an interpretability technique in practical applications, when dealing with other non-synthetic data sources. Noise robustness could be improved at many levels. For instance, generative methods for increasing the density of the state cloud by creating new exemplars might improve the accuracy with which the underlying sampled distribution is represented. Alternatively, methods for improving the numerical stability of obtaining conceptors from a given state cloud could be devised. Such techniques appear particularly relevant when employing relatively low aperture values for the learned conceptors, as they typically contain extremely low values as the conceptor matrix approaches the zero matrix.

4.2.2 Improved graph optimization robustness

The graph optimization process is the final step of NSC, and is responsible for actually generating the output knowledge graph given a host of constraints, including the *expressed abstraction* and *arc density*, among others. There are several hyperparameters which need to be specified in order to properly define the objective function used by the graph optimization procedure. For instance, there are the four coefficients to be found in the linear combination which defines the objective: α, β, γ, and δ. Besides, there are the two hyperparameters which help specify the constraints on local graph structure: $target\_children$ and $target\_parents$. Additionally, there are the few hyperparameters which help specify the underlying simulated annealing algorithm employed: the number of epochs and the temperature schedule used. Finally, the conceptor aperture needs to be specified in order to be able to obtain conceptors from state clouds.

In the context of the present work, all those values have been specified manually. Those have proven quite brittle, especially with larger numbers of concepts to relate to each other in the resulting knowledge graph. Beyond toy examples only containing a handful of concepts, it is extremely difficult to reach sensible outputs based on default hyperparameter values. Given this state of affairs, it would be useful to investigate systematic searches over hyperparameter space, so that suitable values for the graph optimization procedure can be automatically identified. The objective function of this meta-level search for hyperparameters of the graph optimization process can be informed by whether or not the process can successfully recover a few assumed relations of abstraction deemed true (e.g. "fruit" ¿ "apple").

4.2.3 Beyond linear conceptors

Non-linear variations of classical conceptors might be used to capture the internal structure of high-dimensional state clouds of contextual embeddings better than the original elliptical objects. It had been suggested that shallow MLPs could be employed for this task, yet this brings additional questions of interpretability. As we attempt to model high-dimensional representations ever more accurately, we might be forced to depart from sparse human-grounded explanations, an issue resembling the adequacy-fluency trade-off in machine translation.

However, an issue arises when addressing the spatial intuition and theoretical grounding of the abstraction ordering of two conceptors. While this notion is well-defined for linear conceptors (i.e. the ones employed in this work), it is currently an ambiguous notion when it comes to non-linear conceptors. In linear conceptors, abstraction ordering reduces to an evaluation of whether the difference matrix between two conceptors is positive definite. In practice, due to the high amount of noise involved in non-synthetic datasets, we resort to merely approximating this property, by using heuristics based on the eigenvalues of the difference matrix. In non-linear conceptors potentially making use of non-trivial neural networks, the expression of abstraction ordering also becomes non-trivial – the subject of future work.

4.2.4 Beyond abstraction ordering

While this work focuses solely on the meronymous IS_A relationship of abstraction between to concepts, it's conceivable that the difference matrix computed in NSC as an intermediate step contains a rich representation of other relationships between concepts. This is reminiscent of the seemingly algebraic properties of prototype word embeddings (e.g. "king" + "woman" - "man" "queen"), which appear to encode diverse analogies and conceptual relationships. The relative spatial layout of entire state clouds might provide similar information.

However, mathematical objects depicting the aggregate shape of high-dimensional space clouds are likely to capture more information about the semantics of the symbols involved, compared to more rudimentary single prototype embeddings. Assuming the benefit of employing more expressive representation which stays faithful to large sets of exemplars at once, the difference matrices obtained from pairs of conceptor objects might also represent more nuance compared to the less expressive vector of displacement between prototype embeddings.

4.2.5 Beyond token embeddings

The present work focuses solely on extracting a part-whole knowledge graph from the state clouds comprised of contextual embeddings of text tokens (e.g. "juice") or short sequences of such tokens (e.g. "orange juice"). In section 1, however, we mention that NSC is modality-agnostic, in that it can theoretically be employed to distill knowledge graphs from embeddings of other modalities. One possible way of accomplishing that is by pooling together the data points associated with a certain class in an image classification setting. The class would then be identified with a concept based on its label (e.g. "fruit"), while its associated state cloud would not consist in a set of contextual embeddings, as in the case of text, but would be composed of a set of image embeddings generated for entire images (e.g. thousands of images of fruits). By learning conceptors from those state clouds, relating them using abstraction ordering heuristics, and searching for an appropriate knowledge graph structure, it is likely that meaningful meronymous structures would arise. Similar approaches could be employed for yet other modalities. For instance, audio classification datasets would give way to ontologies of audio concepts. Alternatively, datasets comprising video or 3D point clouds, together with labels unifying them in meaningful classes, might also lead to XAI tools in those other modalities.

4.2.6 Alternative abstraction heuristics

In the present work, we employ a rather specific heuristic for quantifying the relation of abstraction between two conceptors. For completeness, we mean-pool the eigenvalues of their difference matrix. Large positive values indicate the first conceptor is deemed to be more abstract than the second, while large negative values indicate the reverse. Values close to zero indicate a limited relation of abstraction between the two.

However, there might be other – better mathematically-motivated – ways of quantifying this relation. For instance, one might opt for different pooling mechanisms (e.g. median), which are more robust against outliers. Alternatively, one might eliminate the eigenvalue pooling mechanism entirely, and focus instead on the *ratio* between positive and negative eigenvalues.

4.2.7 Eliciting latent knowledge

We also note the relevance of the interpretability technique introduced in the present work with respect to conceptual alignment research in AI safety. Researchers in this field have highlighted the relevance of developing interpretability techniques which specifically address the hidden representations employed by DL models. This family of XAI techniques is contrasted with ones which aim solely to relate inputs to outputs when those two pieces of information are expressed in modalities which people are particularly fluent in (e.g. text, images, videos, etc.) as opposed to high-dimensional embeddings.

The motivation behind pursuing interpretability techniques which focus on hidden representations, rather than input and output ones, is rooted in the need to avoid deceptive DL model behavior. Concretely, it has been posited that DL models which internally represent a certain world model could, given the optimization pressure exercised by backpropagation, output predictions which are at odds with the DL model's latent knowledge. For instance, a recent thought experiment portrays a situation in which a DL system is responsible for detecting the theft of a diamond. The associated challenge is to develop mathematical means of eliciting such latent knowledge (ELK) which provably avoid deceptive outputs. By proposing a novel XAI technique which targets hidden states specifically, we see its potential developments as conducive towards addressing the ELK problem.

References