On the Representations of Entities in Auto-regressive Large Language Models

Anonymous ACL submission

Abstract

Better understanding how LLMs manipulate knowledge is both important to design new architectures and control their behavior. While many studies have looked at how relations are expressed, their definition of an entity representation remained naive. In this work, we address the fundamental challenge of associating LLM's internal representations to specific named entity mentions (e.g. Eiffel Tower) within a given context. From a given representation, we explore the extent to which the associated entity mention can be accurately generated. Our experiments demonstrate that the proposed method allows an accurate generation under certain conditions we examine in detail. We also propose simple but effective means to compute entity representations and further validate our results by showing that these representations are linked by relation-specific linear transformations, extending previous work.

1 Introduction

Large Language Models (LLMs) have achieved remarkable success across a range of natural language processing tasks. Despite these achievements, the mechanisms underlying their ability to represent and manipulate knowledge during inference remain opaque, making it challenging to enhance their interpretability and control.

Recent studies have provided insights into how LLMs encode associations between entities and store factual knowledge in their internal representations (Geva et al., 2021, 2023; Hernandez et al., 2024; Niu et al., 2023). For instance, Geva et al. (2023) found that factual associations, such as subject-attribute mappings, can be traced back to the lower layers of transformer networks. A common assumption is that the final token of an entity's mention captures its semantic information.

A key challenge is understanding how LLMs' internal representations can be associated with specific named entities (e.g., "Eiffel Tower") in a given context. Addressing this challenge is crucial for advancing applications such as Named Entity Recognition (NER) or relation extraction. Traditional knowledge representation methods, like Word2Vec, demonstrated that structured information could be projected into continuous vector spaces. While LLMs have the potential to extend this concept by encoding dynamic and context-sensitive representations, current research has focused on entity relationships rather than the internal representation of entities themselves. There is a gap in understanding how well LLMs capture individual entity meanings beyond simple relational knowledge.

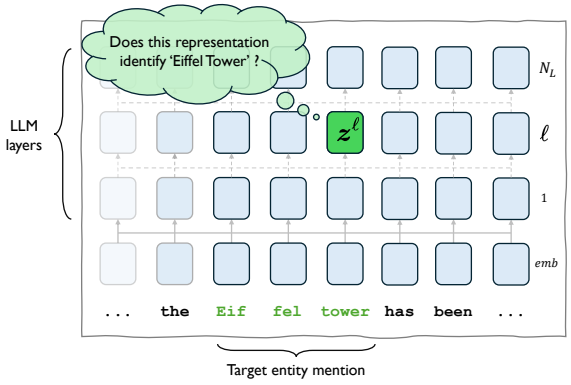


Figure 1: Representation extraction principle: the representation $\boldsymbol{z}^\ell$ (in green) of the last token of a target entity mention - here *tower* for the named entity *Eiffel tower* - is extracted at a given layer ℓ of the transformer model.

In this work, we go further and posit that LLMs rely on layer-agnostic entity representations that can be isolated and manipulated. Our main research question is to associate entities, and more specifically their mentions, with a given representation. We choose to assess the quality of the association between an entity and a given representation by *measuring the extent to which the corresponding mention can be generated from its representation*.

The core of our methodology is presented in Section 3. Section 4 addresses the following question:

RQ 1. *How well entity mentions can be regener-*

ated (decoded) from their representation?

We analyze whether reconstruction depends on the model's layer depth. Our results confirm previous findings (Meng et al., 2022) that middle layers are the best representation of an entity. We also explore whether the decoding results depend on the entity length and frequency and show that its frequency has more impact that the number of tokens it is composed of.

In Section 5 we analyze where the representation is built inside the LLM. Considering successive representation and causal analysis, we show there is no clear location where the entity is "completed".

Our next research question challenges the assumption of using the last token's representation at a given layer. We address it in Section 6.

RQ 2. *Can we find better representations than last token representations from LLMs?*

We found that averaging the representations of the entity's tokens as well as cleaning the representation thanks to a linear transformation help in decoding it more accurately.

Our last research question relates to the additional knowledge entity representations capture; it is developed in Section 7 and is phrased as follows:

RQ 3. *Does the structure of the entity representation space capture (relational) knowledge?*

By transforming representation of subject to related object entities with basic linear transformations, we show that we can extract more than the sole entity mention from those representations. In this section, we also show a practical application of our method which we call the *Entity Lens*, and that extends *logit lens* to multi-token predictions.

2 Related Work

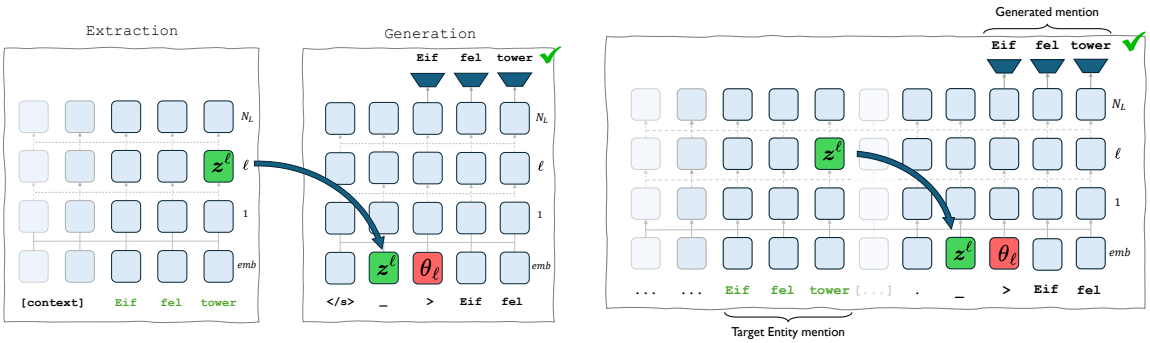
The question of knowledge representation has been studied very early in the development of neural-network approaches for language modeling. Word2Vec (Mikolov, 2013) show that relationships (e.g. singular to plural, capital to country) between words could be represented as translations between word representations. This in turn motivated the development of knowledge base representations where entities were linked by geometrical transformation – the simplest being TransE (Bordes et al., 2013) that re-uses the translation from Word2Vec.

With the development of contextual word – and then token — embeddings, such as ELMo (Peters et al., 2019) and BERT (Devlin, 2018), the question of how knowledge was encoded and how entity were represented has been temporarily put aside, while probing methods were used to determine whether specific types of information (e.g., syntactic structure or factual content) can be extracted from the model's representations (Conneau et al., 2018; Tenney, 2019). These probes are however not fine-grained and do not allow to understand how knowledge is processed and represented.

To get some insight on how LLM process knowledge, a specific type of probe, called "logit attribution" or "logit lens" have been developed (nostalgebraist, 2020; Yu et al., 2023; Dalvi et al., 2019). With logit lens, the LLM hidden states are projected onto the vocabulary using the unembedding matrix, allowing to identify which tokens are the most probable at a given layer of the LLM. Our work can be seen as an extension of logit lens since it can associate a hidden representation with a sequence – the entity mention.

Going back to the question of knowledge, early works (Petroni et al., 2019) showed that pre-trained LLMs can act as "knowledge bases" by retrieving factual information through prompt-based queries. Towards understanding how this retrieval takes place, (Geva et al., 2021; Meng et al., 2022) hypothesized that feed-forward layers in LLMs function as key-value memories, storing and retrieving factual associations during inference. Even if this behavior has has been challenged in some works (Niu et al., 2023), Hernandez et al. (2024) have further shown that within a LLM, the process that transforms the source entity into the target entity with respect to a relation could be approximated by a linear transformation of the representation of the source entity at some middle layer of a decoder – somehow getting back to the observation of (Mikolov, 2013) in the context of LLMs, and suggesting that the LLMs operate at least partially in a space where entities can be represented and manipulated.

This poses the question of how entities are represented. Many works indicate that earlier layers tend to capture surface-level details, while deeper layers encode more abstract and task-specific features (Jawahar et al., 2019; Voita et al., 2019; Geva et al., 2023). This layered approach to information refinement suggests that while entities are represented throughout the network, their representations may not always be coherent or robust, particularly for multi-token entities.



(a) Uncontextual entity mention decoding: The entity representation $\boldsymbol{z}^\ell$ at layer ℓ is extracted in context (left, green), the entity mention is then generated using a learned *task vector* θ_ℓ (right, red) that prompts the model to generate it from $\boldsymbol{z}^\ell$ only.

(b) Contextual entity mention decoding: The entity mention is generated using both the extracted representation $\boldsymbol{z}^\ell$ and the surrounding context.

Figure 2: Comparison of our entity mention decoding method in both uncontextual and contextual senarii.

Representing multi-token entities, such as "Eiffel Tower," indeed poses specific challenges for LLMs. When tokenized into subword units (e.g., `['Eif','fel','_tower']`), the internal representation may not consistently capture the full meaning of the entity. Traditional probing methods that rely on projecting the final token's state onto the vocabulary space often fail to encapsulate the semantics of the entire multi-token mention. This limitation is echoed in the work of (Liu, 2021), who found that tokenization granularity can significantly affect the quality of representations for multi-word expressions. To address these challenges, researchers have explored strategies like attention-based aggregation (Clark, 2019), contextual subword pooling (Schick and Schütze, 2020), and token-level masking (Joshi et al., 2020). Our work specifically studies the problem of how to represent and manipulate multi-token entites.

3 Method and Experimental Settings

In transformer-based language models (Vaswani et al., 2017), text is tokenized into a sequence of tokens $(t_0, \ldots, t_n) \in \mathcal{V}^n$ (where $\mathcal{V}$ the vocabulary used by the tokenizer). These tokens are embedded into a sequence of initial *representations* $(\boldsymbol{z}_0^0, \ldots, \boldsymbol{z}_n^0) \in \mathbb{R}^d$, with d the model's representation space dimension. These representations are sequentially passed through the transformer layers: each layer $\ell \in \{1, \ldots, N_L\}$ generates a new series of representations $(\boldsymbol{z}_0^\ell, \ldots, \boldsymbol{z}_n^\ell) \in \mathbb{R}^d$, building on the representations from the preceding layer.

3.1 Decoding

In this contribution, we measure the quality of the entity representation $\boldsymbol{z} \in \mathbb{R}^d$ at hand by trying to generate the mention from which it was extracted. In Section 4, following the literature (Meng et al., 2022; Geva et al., 2023), $\boldsymbol{z}^\ell$ is extracted from the last token of the entity mention at layer ℓ, as shown Figure 1. (Section 6 extends the experiment to other representations.) To generate a mention for the representation $\boldsymbol{z}^\ell$, we insert $\boldsymbol{z}^\ell$ bypassing the embedding layer, and use *Prompt Tuning* (Lester et al., 2021), which involves optimizing a *soft prompt* of an *embedding vector*, $\theta_\ell \in \mathbb{R}^d$, that prompts the model to reconstruct the entity mention based on the representation $\boldsymbol{z}^\ell$. Because this vector is functional and not semantic, we refer to θ_ℓ as a *task vector*, inspired by the work from Hendel et al. (2023).

Uncontextualized Decoding. In this first setting, the model is only given the representation $\boldsymbol{z}^\ell$ and the task vector θ_ℓ as input. It then has to generate the entire entity mention from only one representation, $\boldsymbol{z}^\ell$. This allows us to measure how much information about the entity mention is retained in $\boldsymbol{z}^\ell$. Figure 2a illustrates this procedure.

Contextualized Decoding. In contrast with the previous setting, in contextualized decoding *we add the textual context from which the representation was extracted* before prompting the transformer with the task vector θ_ℓ. The Figure 2b provides an overview of this methodology. In this setting, the model can just copy the mention from the context. This is however not a trivial task as the model must first identify the correct span within the context, *i.e.*

essentially performing targetted NER.

Task Vector Training In both settings, to learn the task vector θ_ℓ we use standard language modeling loss, *i.e* cross-entropy, to maximize the log-likelihood of generating the entity mention, given the representation and the task vector. The parameters are optimized using Adam (Kingma and Ba, 2017). By training a task vector per layer, we can evaluate which layer provides the most accurate entity representations.

3.2 Dataset and Experiment Setup

Data The CoNLL-2003 dataset (Sang and Meulder, 2003) is a widely-used benchmark for the NER task. The dataset provides a diverse set of named entities across various types, lengths and frequencies, making it well-suited for studying entity representation. In the CoNLL-2003 dataset, the majority of entity mentions are tokenized into 2 or 3 tokens, with some entities spanning up to 8-12 tokens (see Figure 3). This diversity in tokenization length allows us to explore the robustness of the models in handling both short and long multi-token entities, a key aspect of this study.

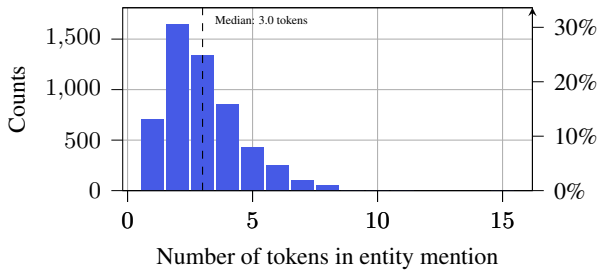


Figure 3: Distribution of the number of tokens for named entity mentions in the test set (PYTHIA tokenizer). 87% of them are tokenized with two or more tokens. The dashed line is the median token count (3).

Models We focus on decoder-only architectures, used in the vast majority of recent models, and known for their superior performance. We experiment with different models from two families. First, the PYTHIA family (Biderman et al., 2023), which includes several models of various sizes, trained with the same data and close settings. This allows to study the impact of architecture parameters on performance, especially the model size. We also use the recent PHI family, a set of models trained with latest techniques and curated data (Li et al., 2023). We use the only two available non-instruct versions, namely PHI-1.5 (Li et al., 2023) and PHI-2 (Javaheripi and Bubeck, 2023), allegedly trained on identical text data, with the particularity that knowledge from PHI-1.5 has been distilled into PHI-2. We also use PHI-3[1] (Abdin et al., 2024), the next iteration of the PHI family. This model has been built following the LLAMA-2 architecture, and has the particularities to have a significantly smaller vocabulary size (32k tokens compared to 50k tokens for all other models considered in this work) and that the only available version of this model is instruction fine-tuned. We use models ranging from 140m to 7B parameters, which offers a fair range for studying the impact of model size, while limiting resource consumption. The architecture parameters of all the the models used in this contribution are detailed in appendix (Table 3).

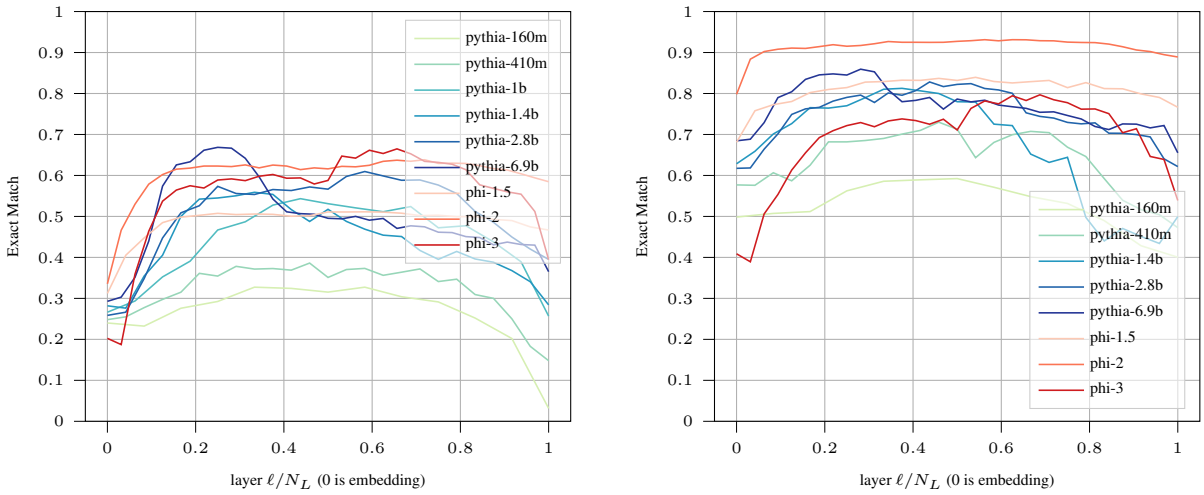
4 Entity mention generation from LLM representations

To address RQ 1, we use multiple models and decode entities from their representation at various layers. We break down the results according to the number of tokens an entity is composed of. We consider both uncontextual and contextual setups.

Layer perspective For uncontextual decoding (Figure 4a), some models (PYTHIA-6.9B, PHI-2 and PHI-3) are able to recover exactly the whole mention of up to of the 65% named mentions. By analyzing the detailed results, we observed that samples that failed often show only small differences with the original mention (see Table 2 in appendix for examples of failures). Figure 4a shows that using representations from the middle layers achieve performance in mention decoding, corroborating findings reported in (Meng et al., 2022, 2023). Model size also unsurprisingly improves the generation performance (e.g. Pythia 160m in green has much lower accuracy than Pythia 6.9b in dark blue). This supports a well known property that larger models memorize more knowledge, including named entities. The best performance as a function of model size is reported in appendix (Figure 9).

These results show that the LLMs are however *in general* not able to retrieve the exact entity mention from the sole representation of its last token at a given layer. In contrast with the sentence embedding models in Morris et al. (2023), auto-regressive language models are not trained to store all the sen-

[1] https://huggingface.co/microsoft/Phi-3-mini-4k-instruct



(a) Uncontextual mention decoding results by layer.

(b) Contextual mention decoding results by layer. Context improves decoding.

Figure 4: Better performances are obtained on representations extracted in middle layers. Curves present the rate of exact match (on y-axis) after training a task vector on representations extracted at layer ℓ (x-axis is normalized layer ℓ/N_L, models having different number of layers N_L).

tence information in their internal representations, since if an entity is already mentioned in the context, the model can access it thanks to the attention mechanism and simply copy it. This is shown in various mechanistic interpretability contributions (Wang et al., 2022; McDougall et al., 2023).

Indeed, we achieve significantly better generation results in the contextual setting. Optimizing a task vector to perform generation with PHI-2 leads to retrieve exactly 93% of the named entity mentions from the representation of their last tokens – was 64% in the uncontextual setup.

Multi-token entities and frequency Decoding the one-token entity `France` from its embedding or representation $\boldsymbol{z}^\ell$ at any layer is far easier than decoding `Mand-el-bro-t` from the sole representation of token `'t'`. To further analyze the impact of mention length on decoding performance, we split the test set by tokenized mention length. Then, following the intuition that an entity mention very frequent in the LLM training set should also be easier to generate than rare ones [2], we also split the test set according to quantiles of mention n-gram frequency in *the Pile* (Gao et al., 2020).[3] We consider both settings for all the models. In both, the frequency of the entity – and not the number of tokens (even if both quantities are obviously correlated) is the main factor for accurate entity mention decoding (See Figure 5 for three PYTHIA models in the uncontextual setting; Appendix D.2 includes the results for all the models and both settings).

Conclusion Put all together, the results from Section 4 bring evidence that LLMs natively bind tokens of a same entity mention together. For named entities that are frequent in the LLM training set, the last token representation at middle layers is enough to generate the whole mention, suggesting that they are almost part of the vocabulary (Figure 5). These representations are however not sufficient for less common entities mentions, suggesting that the model do not store the whole mention in the last token representation (even if an LLM could, see the supplementary experiment in Appendix B.1). Still, when provided with the context, the model can very robustly copy them, achieving up to 93% Exact Matches (EM) (see Figure 4b).

In both settings, our task vector method rightly prompts the model to generate the correct mention from a given representation. This representation can be in turn robustly associated with an entity mention, generalizing the *logit-lens* to multi-token predictions. It also demonstrates that representations are layer-agnostic since injecting them at the embedding level can still be successfully processed by the LLM given a simple task vector.

Despite being similar, the two setups may imply a different generation mechanism from the model. In the uncontexual setup, we require the model

[2]The fact the model fails for rare entities is not a limitation as long as, in their regular operating conditions, LLMs can copy the mention from context.

[3]We got access to the n-gram frequencies thanks to the API proposed by (Liu et al., 2024).

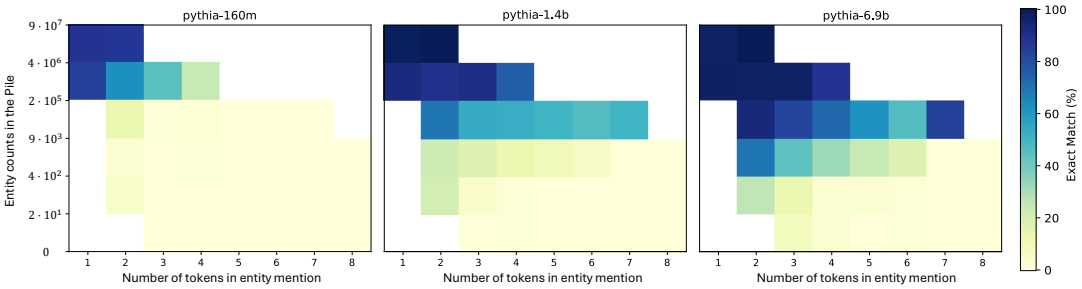


Figure 5: Reconstruction performance for the **uncontextual** mention generation. Performance is separated based on the number of tokens of the entity mention and its n-gram frequency in *the Pile* (Gao et al., 2020). For each model, we chose the layer with best exact match on the test set. Empty cells correspond to count/frequency settings with fewer than 5 samples.

Task Vector	Uncontextual evaluation		Contextual evaluation	
	Exact match	Chr-F	Exact match	Chr-F
Phi-2 ℓ 20, uncontextual	64%	61%	15%	30%
Phi-2 ℓ 20, contextual	41%	40%	93%	94%
Random Vector	0%	13%	0%	17%

Table 1: Despite being trained for distinct tasks, task vectors do show some generalization to other generation settings.

to retrieve a mention from its parametric memory, whereas in the contexual setup we only require it to copy the right mention. Evaluating task vectors across tasks (Table 1) shows that, despite a 50% drop in performance, task vectors trained for one setting can be used in the other. This demonstrates that while the two processes are different, they however have some similarities. Investigating which is left for future work.

5 Analyzing entity representations

Here, we explore how entity mention representations are built inside an LLM.

Successive representations We explore the convergence of the successive representations to the one that we extract at layer ℓ. We consider the cosine similarity of the entity representation $\boldsymbol{z}^\ell$ with the output from different layers, including Multi-head self attention (Attn) and MLP sublayers. Sublayers iteratively *add* or *remove* information in the *residual stream*. There is no predominant layer for the construction of $\boldsymbol{z}^\ell$ (See Figure 6).

Causal analysis is an alternative mean to assess the contribution of one component of a transformer layer on the representation construction. We use *sublayer knockout*, as done in (Geva et al., 2023), by zeroing out the output of one MLP or attention block of a given layer ℓ while computing the representation $\boldsymbol{z}^\ell$. We can measure how much this block

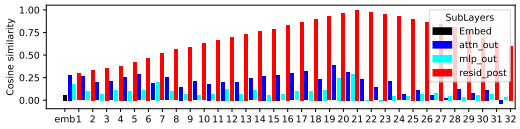


Figure 6: Similarity between the last token representations ($\boldsymbol{z}^\ell$) and intermediate representations from PHI-2. Different sublayers are shown, including outputs from the Multi-Layer Perceptron (MLP) and Multi Head Self-Attention (Attn). Application to a specific prompt : `'Port|ugal| called| up| Port|o| central| defender| Jo|ao| Manuel| P|into'`. The observed behavior is representative of the general one.

contributes to this representation by comparing the similarity of the obtained representation with the original one. We observe that, apart from the first layer, no other blocks have a causal effect on the final representation $\boldsymbol{z}^\ell$, confirming the observation made above.

Conclusion Both our similarity and causal analysis experiments lead us to conclude that, as previous work also suggested (Meng et al., 2022; Geva et al., 2023), there is no clear location where the representation of an entity is "completed". The construction of entity representations are a *a smooth, iterative, and massively superposed process.* This aligns with recent contributions on superposition and feature disentanglement, and may be explained by the use of dropout during LLM pre-training, nudging the model to develop redundant circuits (Elhage et al., 2022; Bricken et al., 2023).

6 Obtaining better representations

We consider two alternative representations for named entities mentions: averaging the entity token representation and cleaning the entity representation.For experiments, we chose PHI-2 model because it is newer compared to PYTHIA, has a

reasonable number of parameters allowing fast inference, and got the best results (See Figure 4).

Average representations The most common way to extract representations of sentences with language models is to average the representations of all their tokens (Jurafsky and Martin, 2009). As superpositions are the building blocks of LLMs (Elhage et al., 2022), using the average entity token representation seems a natural choice. Formally, for a given named entity mention $e = (t_1, ...t_{N_e})$, we compute its average representation at layer ℓ over the entity tokens, $\bar{\boldsymbol{z}}^\ell$ as $\bar{\boldsymbol{z}}^\ell = \frac{1}{N_e}\sum_{i=e_1}^{e_2} \boldsymbol{z}_i^\ell$.

Training a linear layer to *clean* the entity It is well known that representations are highly dependent on the context (Ethayarajh, 2019). In other words, for a given entity mention, the representations extracted from the inner layers of transformer models are very likely to contain, along with the representation of the named entity,a lot of noise that comes from the context. Passing the extracted representation $\boldsymbol{z}$ into a linear layer that would clean or reinforce some relevant features is therefore likely to produce entity representations of better quality, in the sense that the model could better decode the entity mention from them. For both uncontexual and contexual settings, we therefore train a linear model with parameters (W, b) that is applied to the extracted representation $\boldsymbol{z}$ to obtain a *cleaned* representation $\mathcal{C}_{\boldsymbol{z}} = W\boldsymbol{z} + b \in \mathbb{R}^d$

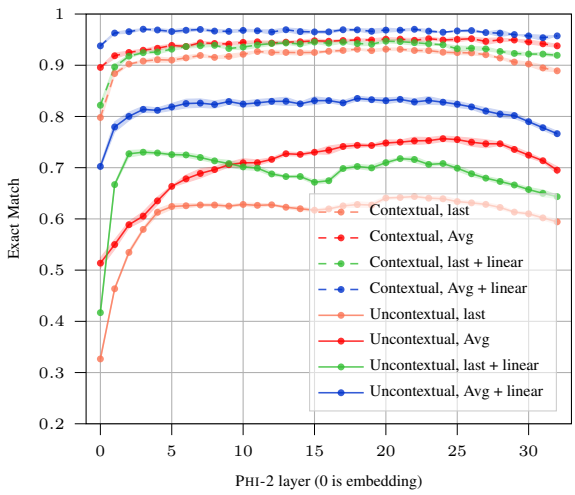


Figure 7: Mention generation with different representations. The representation is extracted as described in Section 3 or by averaging mention representations (Avg) as detailed Section 6. Each experiment was conducted 5 times to get an estimate of the variance.

Discussion Overall, we see that in both uncontexual and contexual setups, the average representation of the tokens from a named entity mention allows to better decode it than the sole representation of the last token (See Figure 7 comparing the red curves with orange ones). That was expected since here all tokens are encoded in the representation, although they are superposed and may thus interfere. The gain compared to representations of last token is aligned with the conclusion of our uncontextual mention generation experiment (see Figure 4a), suggesting that the representation of the last token in auto-regressive LLMs does not *in general* contain all information from all the tokens composing the entity mention.

In the embedding layer (layer 0 in Figure 7) representations $\bar{\boldsymbol{z}}^0$ are just the average of embeddings. For uncontextual decoding, performance jumps to 51% compared to the 33% of exact matches obtained when generating only from the embedding of the last token (solid red and orange curves) – this jump also occurs for contextual decoding, but to a lesser extent. If the model can disentangle all the tokens from their superposed representation in $\bar{\boldsymbol{z}}^0$, it still has to figure out their order to retrieve the original mention.[4] Upper layers representations may therefore encode a notion of relative token position. The fact that last token representations from middle layers yield better results than the average of the token embeddings (up to 64% vs 51%) shows that there is more than only token superposition in the representation of an entity mention.

Transforming the representation with a linear model before inserting it at the embedding layer also improve the reconstruction performance in both settings (Green and blue curves in Figure 7). The generation performance gain means that removing or boosting some relevant features – probably associated with non-entity representation subspaces, *common to all entity representation*, helps the model in generating the correct entity mention.

Overall, this demonstrates that using the sole last token representation at a given layer may not be the best choice when representing an entity. Further work is needed though to understand precisely the effect of the linear transformation and of averaging.

7 Generalizing relation decoding and logit lens

Here, we study a complementary question on whether entity representations can be manipulated. (RQ 3) Recent work on the explainability of knowl-

[4]For instance, "ITALY" is decoded as "YALIT"

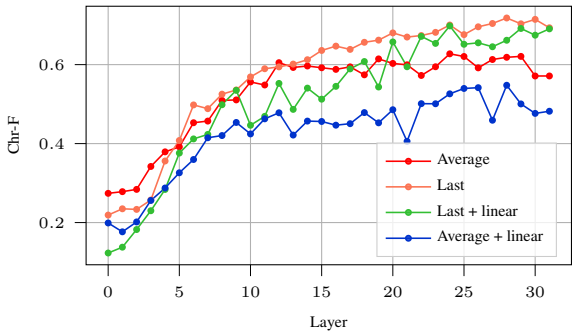


Figure 8: Linear relation decoding on the `Landmarks_to_country` dataset (links landmarks to their home country, e.g `Eiffel tower` and `France`). Averaging or cleaning the representations degrades the semantic information of raw representations from the last token of the entity mention

edge manipulation in transformers (Meng et al., 2022, 2023; Geva et al., 2023; Hernandez et al., 2024; Gottesman and Geva, 2024) support the hypothesis that, for a relation r linking a subject entity s and an object entity o, the representation of the object $\boldsymbol{z}_o$ can be extracted from the representation of the subject $\boldsymbol{z}_s$ in the transformer.

We reproduce the idea from Hernandez et al. (2024) that for some basic relations, the association between $\boldsymbol{z}_s$ and $\boldsymbol{z}_o$ can even be approximated by a linear model. Our work extend those results by properly accounting for multi-token entities. We thus train a linear model $\mathcal{L} : \boldsymbol{z}_s \mapsto W\boldsymbol{z}_s + b$ that aims to project the subject representation $\boldsymbol{z}_s$ on the object representation $\boldsymbol{z}_o$, i.e. $\mathcal{L}(\boldsymbol{z}_s) \approx \boldsymbol{z}_o$.

Experimental settings We use datasets from Hernandez et al. (2024)[5] for which there are enough samples to train our model. The data is filtered to keep only samples for which the relation is encoded into the model's parametric memory, guaranteeing that the object entity is represented somewhere. We then optimize our linear model $\mathcal{L}$ parameters (W, b) using mean-square error $|\mathcal{L}(\boldsymbol{z}_s) - \boldsymbol{z}_o|^2$ on 50 training samples with stochastic gradient descent. We perform this procedure for each layer ℓ, using the different possible representations ($\boldsymbol{z}^\ell$,$\bar{\boldsymbol{z}}^\ell$, $\mathcal{C}_{\boldsymbol{z}^\ell}$ or $\mathcal{C}_{\bar{\boldsymbol{z}}^\ell}$) for subject and object entities. This allows one to see which representations bear the most semantic information about an entity, as well as assessing the quality of the representation from another point of view.

Results Using our mention generation method (See Section 3) allows us to generate a mention for the obtained representation and therefore use *exact match* as a metric instead of only comparing the tokens retrieved using the *logit lens* as done in previous work (Hernandez et al., 2024; Geva et al., 2023). In practice, we present the results using the Character F-score (Chr-F) metric, which is less binary and produces smoother plots. Figure 8 shows that, training a linear model on only 50 samples leads to over 72% Chr-F (74% Exact match) on the `Landmarks_to_country` test set. We show the application to two other relation datasets in appendix (Figure 12). Moreover, if using average or cleaned representations yields better performance on mention decoding, this seems to be at the price of losing semantic information, particularly when it comes to encoding relations to other entities – as shown by the fact that averaging or cleaning entity representations degrades the results.

Entity Lens Finally, we can generate a mention from representations of any token in a text thanks to the task vector. This allows visualizing what entity the model is "thinking" about when processing a token. We introduce the *Entity Lens*, generalizing the *logit lens* (nostalgebraist, 2020) to associate multi-token mentions with any given representation, using the learned task vectors of Section 4. Applying the *Entity Lens* to the sentence "`The City of Lights most iconic monument is the`" for instance shows that the model associates the mention "`The City of Lights`" with Paris, and retrieve a representation associated with "`Eiffel Tower`" while processing the end of the last token "`the`". We present in appendix (Figure 13) entities decoded from various hidden representations.

8 Conclusion

In this paper, we show we can extract layer-agnostic entity (mention) representations from LLMs, decode them using a simple task vector, and transform them for relation decoding, supporting the existence of an entity representation space within LLMs. We have studied two different settings – namely, the contextual and uncontextual ones, and have shown that middle-layer representations of the last token of entity mentions, or their transformations, allow to reconstruct entity mentions. Overall, while there are still some topics that need further investigation, our paper is a first step towards demonstrating the existence of an entity representation space within LLMs.

[5] datasets are available at https://lre.baulab.info/data/

9 Limitations

The prism of mention decoding We consider a vector $\boldsymbol{z} \in \mathbb{R}^d$ to be a good representation of an entity *if* the model is able to generate the correct mention from it. This metric for representation quality is well defined in the contextual case where the correct mention is accessible in the context. It is however ambiguous in the uncontextual case, where an entity may have different acceptable mentions. For instance, in the contextual example of the *Entity Lens* (in Appendix Figure 13) the model generates `Paris` for the representation of the mention `City of Lights`, while this is incorrect according to our evaluation criteria, this result could have been considered as accurate since it shows that the model understood that Paris was the underlying entity.

Generalization of task vectors Our method also relies on the assumption that the representation is layer-agnostic (i.e. the LLM operates with a single representation space). More importantly, we rely on the assumption that the task vector (the only learned parameters) is just enough to instruct the LLM to decode the entity mention without providing any further information.

This discrepancy between contextual and uncontextual settings is underlined by the fact that the learned task vectors are not the same at each layer (see Figure 11 in appendix), although we see that they tend to generalize well to other layers (see in appendix, Figure 14). This choice was motivated by the known performance from *prompt tuning* (Lester et al., 2021), which successfully train embedding vectors to prompt the model into doing specific tasks.

Entity Lens If we extend logit lens by allowing a multi-token generation, we are limited to generating only one mention, whereas the logit lens allows to retrieve the top-k mappings. This limitation could however be solved, for instance by performing beam-search generation, allowing to get the "top-k" entity mentions for a given representation.

References

M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. Awadalla, N. Bach, A. Bahree, A. Bakhtiari, H. S. Behl, A. Benhaim, M. Bilenko, J. Bjorck, S. Bubeck, M. Cai, C. C. T. Mendes, W. Chen, V. Chaudhary, P. Chopra, A. D. Giorno, G. de Rosa, M. Dixon, R. Eldan, D. Iter, A. Goswami, S. Gunasekar, E. Haider, J. Hao, R. J. Hewett, J. Huynh, M. Javaheripi, X. Jin, P. Kauffmann, N. Karampatziakis, D. Kim, M. Khademi, L. Kurilenko, J. R. Lee, Y. T. Lee, Y. Li, C. Liang, W. Liu, E. Lin, Z. Lin, P. Madan, A. Mitra, H. Modi, A. Nguyen, B. Norick, B. Patra, D. Perez-Becker, T. Portet, R. Pryzant, H. Qin, M. Radmilac, C. Rosset, S. Roy, O. Saarikivi, A. Saied, A. Salim, M. Santacroce, S. Shah, N. Shang, H. Sharma, X. Song, O. Ruwase, X. Wang, R. Ward, G. Wang, P. Witte, M. Wyatt, C. Xu, J. Xu, S. Yadav, F. Yang, Z. Yang, D. Yu, C.-Y. Zhang, C. Zhang, J. Zhang, L. Zhang, Y. Zhang, Y. Zhang, and X. Zhou. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. Apr. 2024.

S. Biderman, H. Schoelkopf, Q. Anthony, H. Bradley, K. O'Brien, E. Hallahan, M. A. Khan, S. Purohit, U. S. Prashanth, E. Raff, A. Skowron, L. Sutawika, and O. van der Wal. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling, May 2023.

A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems*, 2013. URL https://api.semanticscholar.org/CorpusID:14941970.

T. Bricken, C. Olah, and A. Tempelton. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. Oct. 2023.

K. Clark. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

A. Conneau, G. Kruszewski, G. Lample, L. Barrault, and M. Baroni. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*, 2018.

F. Dalvi, N. Durrani, H. Sajjad, Y. Belinkov, A. Bau, and J. Glass. What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019. ISBN 978-1-57735-809-1. doi: 10.1609/aaai.v33i01.33016309. URL https://doi.org/10.1609/aaai.v33i01.33016309.

J. Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

N. Elhage, T. Hume, C. Olsson, N. Schiefer, T. Henighan, S. Kravec, Z. Hatfield-Dodds, R. Lasenby, D. Drain, C. Chen, R. Grosse, S. McCandlish, J. Kaplan, D. Amodei, M. Wattenberg, and C. Olah. Toy Models of Superposition. 2022. doi: 10.48550/ARXIV.2209.10652.

K. Ethayarajh. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *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 55–65, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1006.

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, S. Presser, and C. Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020. URL https://arxiv.org/abs/2101.00027.

M. Geva, R. Schuster, J. Berant, and O. Levy. Transformer Feed-Forward Layers Are Key-Value Memories, Sept. 2021.

M. Geva, J. Bastings, K. Filippova, and A. Globerson. Dissecting Recall of Factual Associations in Auto-Regressive Language Models, Oct. 2023.

D. Gottesman and M. Geva. Estimating Knowledge in Large Language Models Without Generating a Single Token. June 2024.

R. Hendel, M. Geva, and A. Globerson. In-Context Learning Creates Task Vectors. 2023. doi: 10.48550/ARXIV.2310.15916.

E. Hernandez, A. S. Sharma, T. Haklay, K. Meng, M. Wattenberg, J. Andreas, Y. Belinkov, and D. Bau. Linearity of Relation Decoding in Transformer Language Models, Feb. 2024.

M. Javaheripi and S. Bubeck. Phi-2: The surprising power of small language models, Dec. 2023.

G. Jawahar, B. Sagot, and D. Seddah. What does BERT learn about the structure of language? In A. Korhonen, D. Traum, and L. Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1356. URL https://aclanthology.org/P19-1356.

M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the association for computational linguistics*, 8:64–77, 2020.

D. Jurafsky and J. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall Series in Artificial Intelligence. Pearson Prentice Hall, 2009. ISBN 978-0-13-187321-6.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

B. Lester, R. Al-Rfou, and N. Constant. The Power of Scale for Parameter-Efficient Prompt Tuning, Sept. 2021.

Y. Li, S. Bubeck, R. Eldan, A. Del Giorno, S. Gunasekar, and Y. T. Lee. Textbooks are all you need ii: **phi-1.5** technical report. *arXiv preprint arXiv:2309.05463*, 2023.

J. Liu, S. Min, L. Zettlemoyer, Y. Choi, and H. Hajishirzi. Infini-gram: Scaling Unbounded n-gram Language Models to a Trillion Tokens. 2024. doi: 10.48550/ARXIV.2401.17377.

W. Liu. Caveats for the use of web of science core collection in old literature retrieval and historical bibliometric analysis. *Technological Forecasting and Social Change*, 172:121023, Nov. 2021. ISSN 0040-1625. doi: 10.1016/j.techfore.2021.121023. URL http://dx.doi.org/10.1016/j.techfore.2021.121023.

C. McDougall, A. Conmy, C. Rushing, T. McGrath, and N. Nanda. Copy Suppression: Comprehensively Understanding an Attention Head, Oct. 2023.

K. Meng, D. Bau, A. Andonian, and Y. Belinkov. Locating and Editing Factual Associations in GPT. *Advances in Neural Information Processing Systems*, 35:17359–17372, Dec. 2022.

K. Meng, A. S. Sharma, A. Andonian, Y. Belinkov, and D. Bau. Mass-Editing Memory in a Transformer, Aug. 2023.

T. Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

J. X. Morris, V. Kuleshov, V. Shmatikov, and A. M. Rush. Text Embeddings Reveal (Almost) As Much As Text. [object Object], 2023. doi: 10.48550/ARXIV.2310.06816.

N. Nanda and J. Bloom. Transformerlens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

J. Niu, A. Liu, Z. Zhu, and G. Penn. What does the Knowledge Neuron Thesis Have to do with Knowledge? In *The Twelfth International Conference on Learning Representations*, Oct. 2023.

nostalgebraist. Interpreting GPT: The logit lens. Aug. 2020.

M. E. Peters, M. Neumann, R. L. Logan IV, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019.

F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

E. F. T. K. Sang and F. D. Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition, 2003. URL https://arxiv.org/abs/cs/0306050.

T. Schick and H. Schütze. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*, 2020.

I. Tenney. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. ukasz Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

E. Voita, R. Sennrich, and I. Titov. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *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 4396–4406, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1448. URL https://aclanthology.org/D19-1448.

K. Wang, A. Variengien, A. Conmy, B. Shlegeris, and J. Steinhardt. Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 small, Nov. 2022.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.

Q. Yu, J. Merullo, and E. Pavlick. Characterizing Mechanisms for Factual Recall in Language Models, Oct. 2023.

A Reproducibility statement

To implement our experiments, we used the Transformer Lens (Nanda and Bloom, 2022), a wrapper around the transformer library (Wolf et al., 2020). The code will be published upon publication. All experiments were conducted on workstations with 80GB NVIDIA A100, 16 or 32GB NVIDIA V100 SXM2-HBM2 GPUs. The selected datasets used in our experiments Section 7 will also be listed, in addition to the one already mentioned in Section 3.2.

B Additional experiments

B.1 Training Representations

We explore in this section what features in the representation are really used to generate a mention. To obtain the minimum information required to retrieve the mention, we optimize blank noise to make the model retrieve the right mention when prompted with a task vector trained in the context of our uncontexual mention generation experiment. By doing it several times and averaging the obtained vectors, we hope to keep only hat is needed to regenerate the mention.

Conclusion Our results show that we can train a vector to contain almost any mention (up to a reasonable number of tokens).

C Textual Examples

D Complementary Figures

D.1 Entity Lens visualizations

We provide here two example applications of the *Entity Lens*: Figure 13 for the uncontexual setup and Figure 15 in the contexual setup. For any layer ℓ and for each token representation $\boldsymbol{z}_k^\ell$, we generate a mention with the layer-specific task Vector θ_ℓ. To test generalization capabilities through layers, we try Figure 14 to use the same task vector θ_{20} for all the layers, empirically validating nice generalization capabilities.

D.2 Performance as a function of Entity mention size and frequency

Original	Inferred
Roberto Mancini	Carlo Mazzone
Pierre Van Hooydonk	Marc D'Haese
Guenther Huber	Peter Huber
Wenchang	Changsha
Michael Cornwell	Mark Calwell
IGLS	GLIS
Ole Einar Bjorndalen	Bjorn Dæhlie
Alba Berlin	Berlin
John Langmore	John Molyneaux
Patasse	Passeau
Rangoon	Yangon
M. Waugh	J. Waugh
Major	John Major
Lahd	Hlad
WARSAW	WAWRZAWA
Kim Pan Keun	Lee Dong-kook
David Boon	J. Boon
Berisha	Bushi
Gunn Margit Andreassen	Ingrid Bjørnson
Abel Balbo	Giuseppe Bologna

Table 2: Examples of failed generations of the uncontextual mention generation (Figure 4a) for PHI-2 at layer 15.

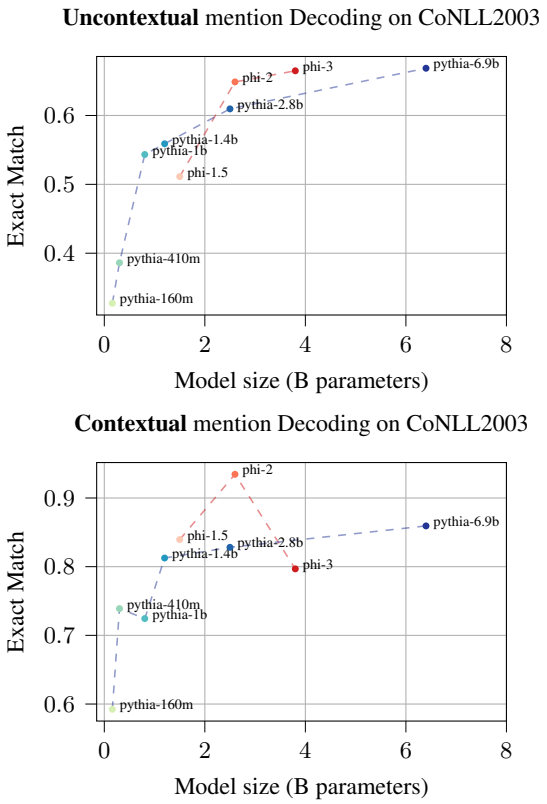


Figure 9: Aggregated Results comparing best performances depending on model size. Larger models demonstrate greater capability. Performance drop of the PHI-3 model can be explained by the use of a significantly smaller vocabulary size (32k vs 50k for all other models considered here)

	n_params	n_layers	d_model	n_heads	act_fn	n_ctx	d_vocab	d_head	d_mlp
PHI-1.5	1.2B	24	2048	32	gelu	2048	51200	64	8192
PHI-2	2.5B	32	2560	32	gelu	2048	51200	80	10240
PHI-3	3.6B	32	3072	32	silu	4096	32064	96	8192
PYTHIA-160M	85M	12	768	12	gelu	2048	50304	64	3072
PYTHIA-410M	302M	24	1024	16	gelu	2048	50304	64	4096
PYTHIA-1B	805M	16	2048	8	gelu	2048	50304	256	8192
PYTHIA-1.4B	1.2B	24	2048	16	gelu	2048	50304	128	8192
PYTHIA-2.8B	2.5B	32	2560	32	gelu	2048	50304	80	10240
PYTHIA-6.9B	6.4B	32	4096	32	gelu	2048	50432	128	16384

Table 3: Characteristics of the models considered in this work.

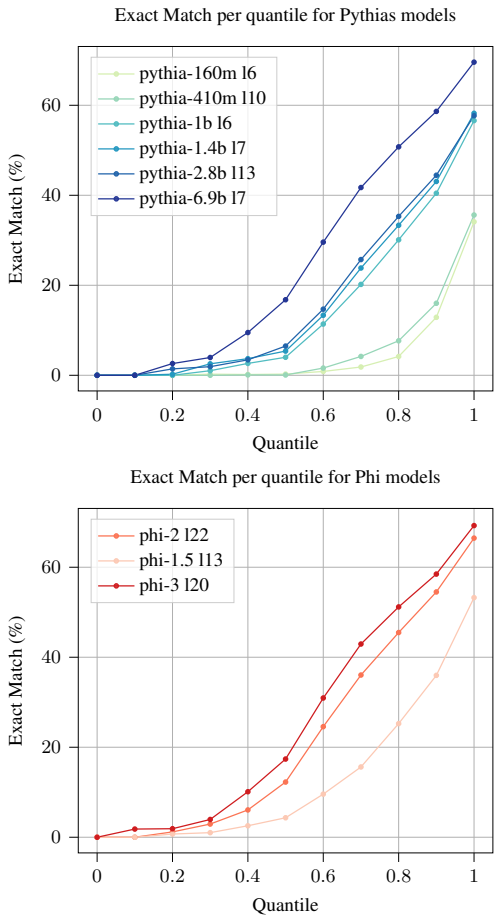
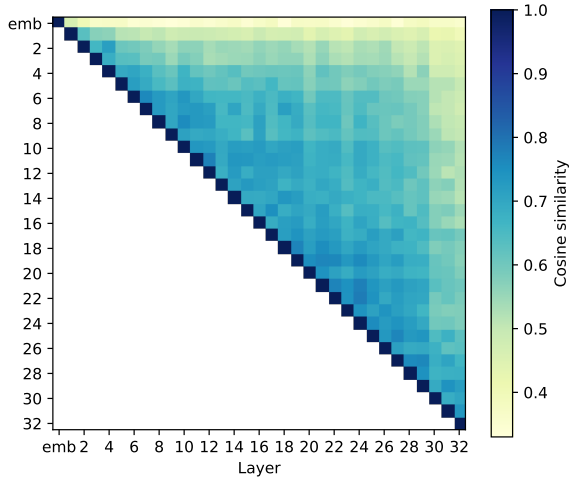
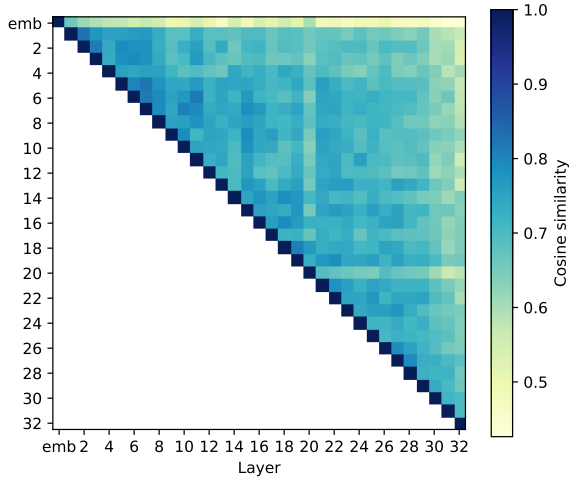


Figure 10: Performance on mention generation without context depending on quantiles of entity frequency in the Pile.

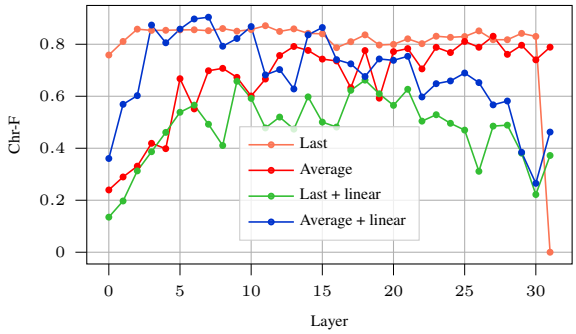


(a) Similarities between trained task vectors in the uncontexual setup

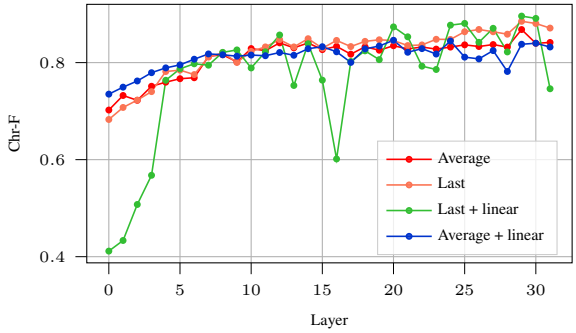


(b) Similarities between trained task vectors in the contexual setup

Figure 11: Cosine Similarity comparison of all trained task Vectors for PHI-2



(a) Performance on `star_constellation`



(b) Performance on `person_native_language`

Figure 12: Chr-F performance on other datasets from Hernandez et al. (2024).

Uncontextual Entity-Lens, TaskVec from all layers

Layer	<|endoftext|>	The	City	of	Lights	most	iconic	monument	is	the
0		Reuters	City	Croatia	Lights	most	iconic	monument	not a good thing.	Reuters
1	Swiss National Bank	The	City	City of	Lights	most	iconic	monument	U.S. Treasury Secretary Henry Paul	U.S. President George W.
6	M	PGA Tour	City	City	City of Lights	most	iconic	most famous monument	MOST VALUED MOST VAL	M
11	Rangers	The	City	City of	City of Lights	most	most iconic	most famous monument	B.	Bank
16		The	City	City of	City of Lights	most	iconic's	monument		
21	C.	The	City	City of	City of Lights	Manchester United	iconic	monument's monument	Mount Everest	Bordeaux
26	M.	The	City	City of	Paris Lights	New York City	iconic	monument		
31	Baltimore	The	City	City of	Paris	Most	most iconic	most famous monument	Paris	Eiffel Tower

Figure 13: The *Entity Lens*, applied with task vectors trained on representations from PHI-2 in the uncontexual setup. An example application in the contextual setup can be found Figure 13. (layer 0 is embedding)

Uncontextual Entity-Lens, TaskVec from layer 20

Layer	<|endoftext|>	The	City	of	Lights	most	iconic	monument	is	the
0	T-Mobile USA Inc.	B.	City	B.C.	Lights	S.C. Braga	iconic	monument	S	G.
1	F.C. Porto	The	City	City	Lights	S.C. Braga	iconic	monument	S.C. Braga	S.C. Braga
6	C.	B.	City	City	City of Lights	Barcelona	iconic	monument	Olympic Games	S.C. Braga
11	C.	B.	City	City of	The City of Lights	S.C. Braga most	city's most iconic	monument	F1	F.C. Porto
16	C.	B.	City	City of	City of Lights	Manchester United	iconic	monument	Mount Everest	U.S.
21	C.	The	City	City of	City of Lights	Manchester United	iconic	monument's monument	Mount Everest	Bordeaux
26	C.A.F.C.	S.	City	City of	City of Lights	Bordeaux	iconic	monument's monument	Notre Dame de Paris	Eiffel Tower
31	F.C. Porto	Oceania	City	City of Edmonton	Paris Light	C.	Eiffel Tower's iconic landmark	Monument de la Liberté	Paris	Eiffel Tower

Figure 14: The *Entity Lens*, applied with only the task vector trained on representations extracted at layer 20 of PHI-2 in the uncontexual setup. We see nice generalizing capabilities of θ_{20}, the model still decodes relevant mentions from represenations extracted at different layers. (0 is embedding)

Contextual Entity-Lens, TaskVec from all layers

Layer	<|endoftext|>	When	Albert	Einstein	and	Mand	el	bro	t	went	to	the	store	,	the	General	Rel	ativity	father	gave	the	bottle	to
0	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot	Mandel	Mandelbrot	Mandelbrot	Mandelbrot	Mandelbrot	Mandelbrot	Mandelbrot	General Relativity father	Mandelbrot	Mandelbrot	General	Rel	Mandelbrot	General Relativity father	Mandelbrot gave	Mandelbrot	Mandelbrot	Mandelbrot
1	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot	Mandel	Mandelbrot	Mandelbrot	Mandelbrot	Mandelbrot went	Mandelbrot	Mandelbrot	Store	Mandelbrot	Mandelbrot	General	Rel	General Relativity	Mandelbrot	Mandelbrot gave	Mandelbrot	Mandelbrot went to the store	Mandelbrot
6	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot	Mand	Mandel	Mandelbrot	Mandelbrot	Mandelbrot went to the store	Mandelbrot went to the store	Mandelbrot went to the store	store	Mandelbrot	Mandelbrot	General	Gen Rel	General Relativity	Mandelbrot father	Mandelbrot gave	Mandelbrot went to the store	Bottle	Mandelbrot went to the store
11	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot	Mand	Mandel	Mandelbrot	Mandelbrot	Mandelbrot went to the store	Go to the store.	Not the other way around	store	Go to the store.	Mandelbrot	General	Rel	Mandelbrot	father	Gave	Mandelbrot went to the store	Bottle	Mandelbrot went to the store
16	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot		Mandel	Mandelbro	Mandelbrot	Mandelbrot went to the store	Go to the store	Mandelbrot go to the store	store	Mandelbrot went to the store	Mandelbrot	General	Gen Rel	General Relativity	Albert Einstein	Gave the bottle to gave	Mandelbrot	Bottle	Mandelbrot go to the store
21	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot father	Mand	Mandelbrot	Mandelbro	Mandelbrot	Mandelbrot went to the store	store	store	store	Mandelbrot went to the store	Mandelbrot	General	Gen Rel	General Relativity	Albert Einstein	Mandelbrot gave	Mandelbrot	Bottle	Mandelbrot went to the store
26	Mandelbrot	Mandelbrot	Albert	Albert Einstein	Mandelbrot	Mand	Mandelbrot	Mandelbrot	Mandelbrot	Store	store	store	Store	Mandelbrot	Mandelbrot	General	Gen Rel	General Relativity	Albert Einstein	Mandelbrot	Mandelbrot	Bottle	Mandelbrot
31	Bottle	Albert Einstein	Albert	Albert Einstein	Mandelbrot	Mandel	Mandelbrot	Mandelbro	Mandelbrot	Store	to	store	Store	Albert Einstein	store	General	General Relativity	General Relativity	father	bottle	Mandelbrot	bottle	bottle

Figure 15: The *Entity Lens*, applied with a task Vector trained on representations extracted in PHI-2 in the contexual setup.(layer 0 is embedding)

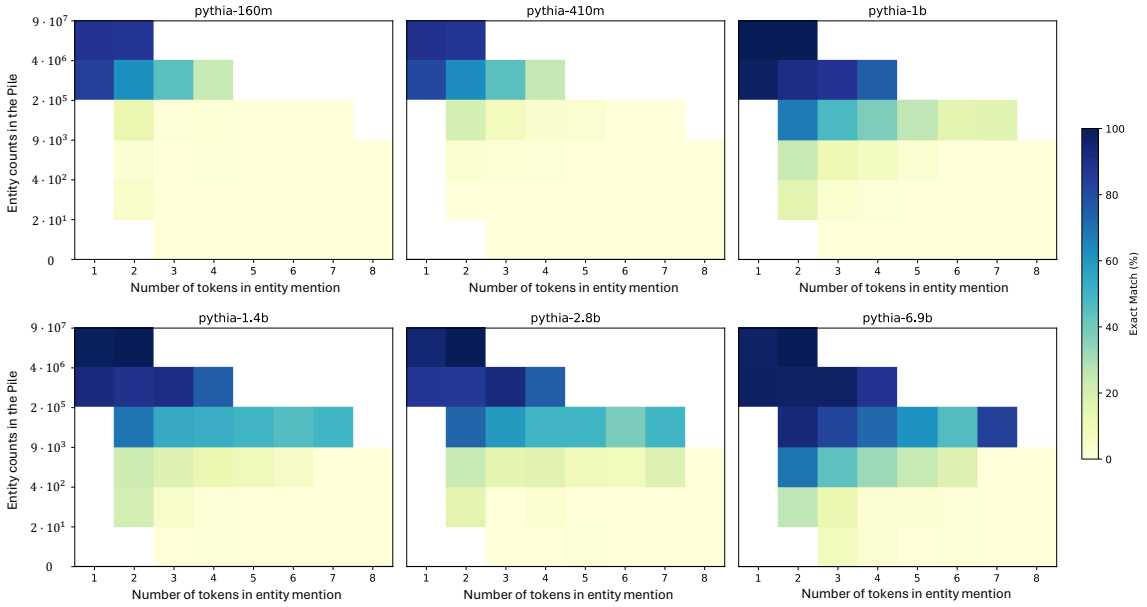


Figure 16: Reconstruction performance on test set for our **uncontextual** mention generation experiment. Performance is separated based on the number of tokens that need reconstruction, as well as the n-gram frequency of the mention in *the Pile* (Gao et al., 2020). For each model, we chose the layer with best exact match on the test set. Empty cells correspond to count/frequency settings with fewer than 5 samples, making it insufficient to compute performance.

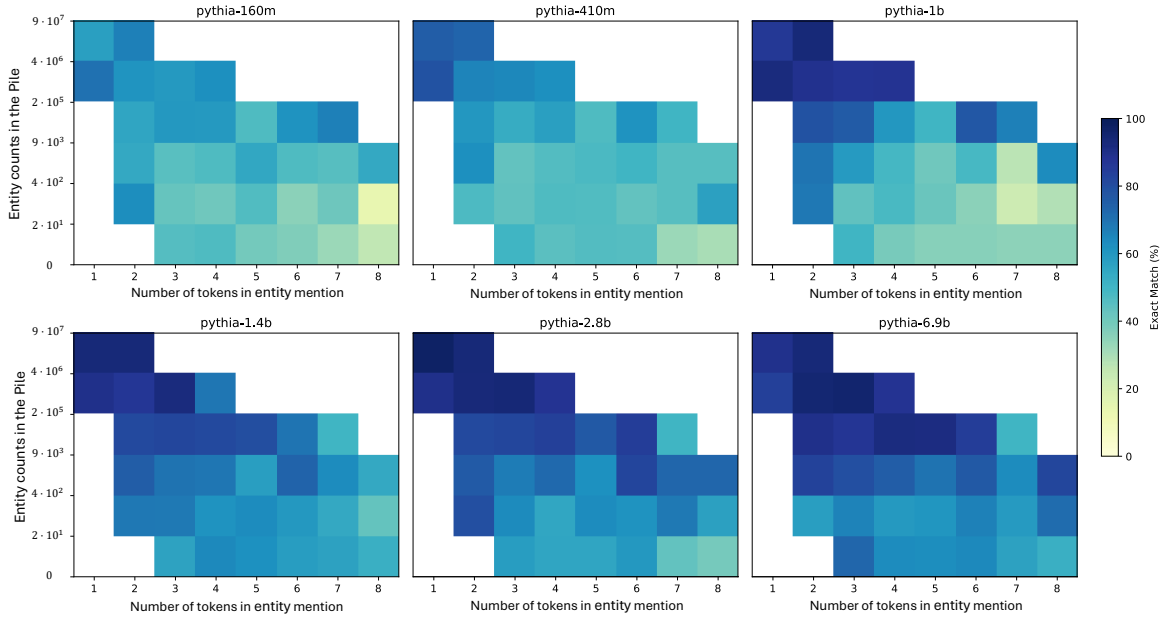
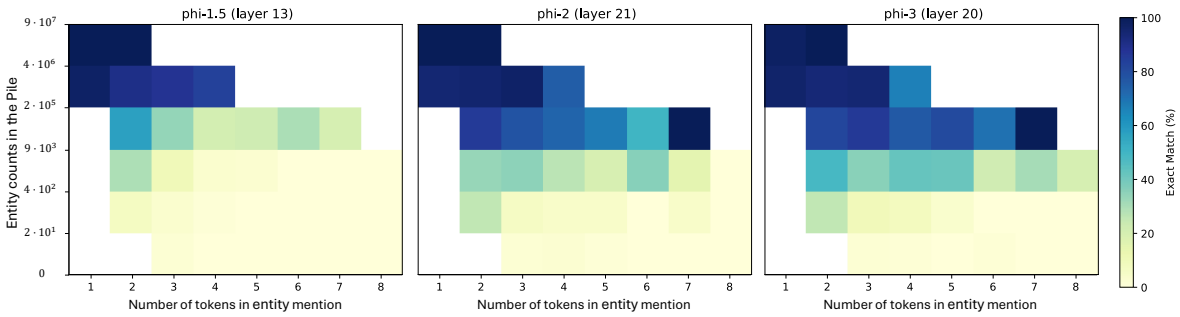
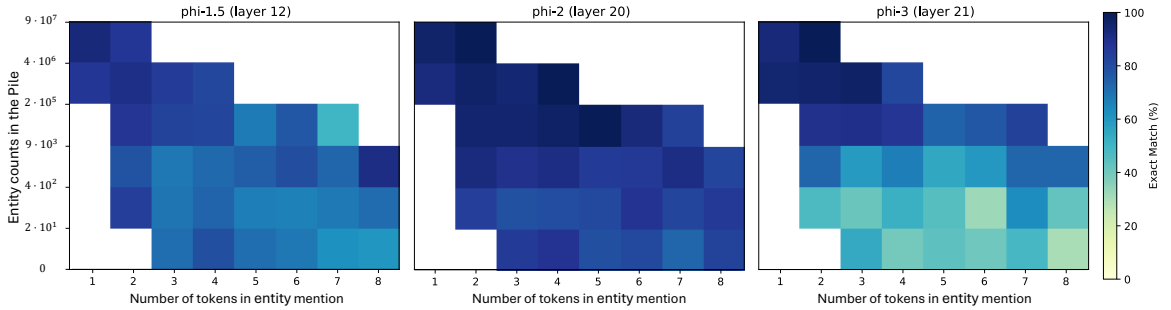


Figure 17: Reconstruction performance on test set for our **contextual** mention generation experiment. Performance is separated based on the number of tokens that need reconstruction, as well as the n-gram frequency of the mention in *the Pile* (Gao et al., 2020). For each model, we chose the layer with best exact match on the test set. Empty cells correspond to count/frequency settings with fewer than 5 samples, making it insufficient to compute performance.



(a) Performance analysis for the uncontextual mention generation experiment



(b) Performance analysis for the contextual mention generation experiment

Figure 18: Reconstruction performance depending on the number of tokens to reconstruct, as well as the n-gram frequency of the entity mention in *the Pile* (Gao et al., 2020). For each model, we chose the layer with best exact match on the test set.