Generalisation First, Memorisation Second? Memorisation Localisation for Natural Language Classification Tasks

Verna Dankers[1] and **Ivan Titov**[1,2]
[1]ILCC, University of Edinburgh
[2]ILLC, University of Amsterdam
vernadankers@gmail.com, ititov@inf.ed.ac.uk

Abstract

Memorisation is a natural part of learning from real-world data: neural models pick up on atypical input-output combinations and store those training examples in their parameter space. *That* this happens is well-known, but *how* and *where* are questions that remain largely unanswered. Given a multi-layered neural model, where does memorisation occur in the millions of parameters? Related work reports conflicting findings: a dominant hypothesis based on image classification is that lower layers learn generalisable features and that deeper layers specialise and memorise. Work from NLP suggests this does not apply to language models, but has been mainly focused on memorisation of facts. We expand the scope of the localisation question to 12 natural language classification tasks and apply 4 memorisation localisation techniques. Our results indicate that memorisation is a gradual process rather than a localised one, establish that memorisation is task-dependent, and give nuance to the generalisation first, memorisation second hypothesis.

1 Introduction

Memorisation in neural models is both concerning due to overfitting and privacy concerns, and desired because of information that needs to be stored, such as facts. With the recent surge in the number of models trained on very large, closed-access training corpora, the NLP community has seen an increased interest in research questions that aim to improve our understanding of memorisation, such as: Which data points are memorised, and can we extract those examples from models (e.g. Chang et al., 2023a; Shi et al., 2023; Nasr et al., 2023)? How dependent is memorisation on model scale, architecture and training procedures (e.g. Carlini et al., 2022)? Can we localise memorised information (i.e. pinpoint which weights, subcomponents or layers are most associated with storing that information) and edit models' memories (e.g. Meng et al., 2022a; Hase et al., 2024)?

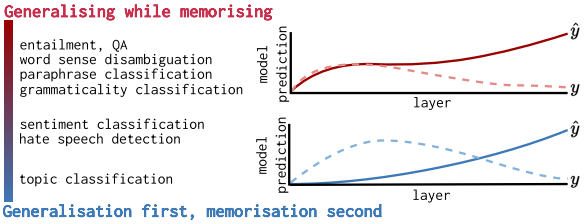


Figure 1: If we train transformer to memorise incorrect label $\hat{y}$, the implementation of that memorisation is task-dependent. We demonstrate this for 12 NLP classification tasks. The visualisation is for illustrative purposes.

Memorisation localisation is the central theme in this work, specifically localisation at the level of layers. There is a lack of consensus about which layers are particularly involved in memorisation in deep neural models, which may stem from the varying experimental setups and varying definitions of memorisation employed by different studies. Work from *computer vision* (CV) mostly focused on memorisation of perfectly memorised mislabelled examples, positing that lower layers capture generalisable features while deeper layers memorise (e.g. Baldock et al., 2021; Stephenson et al., 2021) (although that has been recently challenged by Maini et al. (2023)). Work from NLP often discusses memorisation of facts, for which lower (Geva et al., 2023), middle (Meng et al., 2022a) and final layers (Dai et al., 2022) have all been mentioned as playing crucial roles in *pre-trained language models* (PLMs). And lately, with the availability of some open-access pre-training data, memorisation localisation for sequences memorised verbatim has gained traction, and initial results primarily point to lower layers (Stoehr et al., 2024). Different articles investigating different types of memorisation arrive at different answers, and even work focused on a specific type (such as facts) does not always agree.

We contribute an important piece of the puzzle in the memorisation localisation landscape, by employing a setup that is similar to Stephenson et al. and Maini et al.: we perform layer-wise localisation in fine-tuned models for 12 NLP classification

Findings of the Association for Computational Linguistics: ACL 2024, pages 14348–14366

tasks, enforcing memorisation by applying label perturbation to a data subset. We use four memorisation localisation methods (§3), and first examine their accuracy in a control setup. Afterwards (§4), we address the main research question: **In which layers does memorisation occur?** The results do not always align – which underscores that we should not overly rely on one localisation method – but do tell a coherent story. In §5, we introduce a visualisation technique (centroid analysis) to make this story more interpretable: memorisation is not sudden but gradual. Together, layers gradually shift mislabelled examples to their newly assigned class, but *when* that happens is task-dependent: the better a model generalises to new data for a particular task, the more relevant deeper layers are for memorisation. Figure 1 illustrates this. Our findings beg a nuance of the generalisation first, memorisation second hypothesis, and we end with a discussion (§6) of what our findings mean for localisation and model editing going forward.

2 Related work

In this section, we discuss a broad range of related work on memorisation localisation from CV and NLP, focusing specifically on studies that discuss the role of different *layers* in deep neural networks.

Noise memorisation in CV Multiple image classification studies concluded that deeper layers are more involved in memorisation than earlier layers, based on analyses of the memorisation of mislabelled or hard examples. This has been reported by work contrasting entire models (regular models and ones trained with randomised labels) (Morcos et al., 2018; Cohen et al., 2018; Ansuini et al., 2019, i.a.), and work discussing how some examples are handled differently within one model: Baldock et al. (2021) establish a positive correlation between prediction depth in image classification (the earliest layer that predicts the label) and example-level learning difficulty metrics. Stephenson et al. (2021) analyse hidden representations for image classification and report that memorisation of mislabelled examples occurs abruptly in late layers and late training epochs. Rewinding models' final layers to earlier checkpoints reversed memorisation.

Contrary to previous work, Maini et al. (2023) report that for image classification datasets that were partially mislabelled, instead of memorisation being confined to specific layers, there are small sets of neurons dispersed over the full architecture involved in memorisation.

Memorisation of factual knowledge NLP memorisation localisation studies have primarily focused on factual knowledge, although only a subset of work in this direction discusses the roles of different layers. De Cao et al. (2021) first connected work from CV to fact memorisation in transformer, and train a hypernetwork to edit facts. Their hypernetwork mostly edits the bottom layer of a six-layer transformer, and De Cao et al. suggest this difference might be due to the change in modality.

Later work operated under the assumption of the *knowledge neuron thesis*,[1] assuming that facts are recalled from the training corpus through transformer's MLP weights, that act as a key-value memory (Geva et al., 2021), and that one may thus be able to identify MLP knowledge neurons (Dai et al., 2022): (1) Meng et al. (2022a,b) edit factual memories in transformer-based PLMs' MLPs by first localising memorisation to specific layers and only updating those layers, in which case early/mid-layers are most often selected.[2] Note that Hase et al. (2024) find success in model editing to be unrelated to the layers selected by Meng et al.'s localisation method, which means that model editing might be an unreliable way to check where facts are stored. (2) Dai et al. (2022) identify knowledge neurons for factual information, and mostly find neurons in the *top* layers of `BERT`, with similar findings being reported in later work on knowledge neurons by Zhao et al. (2024) and Chen et al. (2024). Dai et al. then succesfully use those neurons to update facts.

Although the articles above disagree in terms of whether higher or lower layers store factual information, work beyond model editing and knowledge neurons indicates factual information is already present in the lower layers: Haviv et al. (2023) show that facts and idioms are stored and retrieved in early layers in `BERT` and `GPT-2`. Deeper layers perform confidence boosting *after* retrieval. Geva et al. (2023) artificially block parts of the computation in fact retrieval for `GPT-J` and `GPT-2 XL`, and find that factual information is stored in the lower MLP sublayers, which is echoed by recent work from Ortu et al. (2024).

Verbatim memorisation While memorisation localisation has predominantly focused on facts,

[1] The term was coined by Niu et al. (2024) to summarise the hypothesis underlying multiple related studies. Niu et al. criticise the thesis since it oversimplifies knowledge storage. Instead, they suggest to focus on network-wide circuits.

[2] E.g. generalisation to paraphrased prompts of modified facts peaks when editing layer 18 out of 48 for `GPT-2 XL`. This result is not specific to transformer: Sharma et al. (2024) find early/mid-layers to be important when editing facts in `Mamba`.

recent open science initiatives publishing PLM pre-training corpora enable research on memorisation of pre-training data. Chang et al. (2023b) present a dataset to evaluate different localisation methods by relying on text memorised verbatim by three PLMs, but they do not elaborate on the roles of layers. Stoehr et al. (2024) examine for one specific 12-layer architecture (`GPT-Neo-125M`, that we will also analyse) which model components are responsible for memorising sequences of 50 tokens verbatim, finding lower layers to be the most relevant. Localisation of verbatim memorisation is hard to standardise across different tasks and models due to the required access to pre-training data, and because all PLMs memorise different sequences.

Memorisation beyond localisation Other work on memorisation in NLP examines the conditions under which memorisation occurs during fine-tuning (Tänzer et al., 2022; Mireshghallah et al., 2022), which fine-tuning tasks lead to the most memorisation (Zeng et al., 2023), which examples are memorised (Biderman et al., 2024) and when memorisation is beneficial for generalisation (Zhang et al., 2023; Zheng and Jiang, 2022).

Summarising, many different conclusions have been drawn in memorisation localisation studies. It is unclear whether the different conclusions from different articles may be reduced to a difference between the vision and language modalities, to a difference between the different types of memorisation investigated, to localisation techniques employed or even to the varying models and model sizes used in related work.[3] We take away some of that confusion by (1) employing a setup previously used for the vision modality and (2) investigating a type of memorisation ('noise memorisation') understudied in NLP, using widely studied models. This allows us to conclude whether the 'deeper layers' answer from CV really stands in contrast with the 'lower layers' answer from the majority of NLP studies (or whether that was simply unique to noise memorisation), and allows us to investigate whether the 'lower layers' answer is unique to fact memorisation and verbatim memorisation.

3 Methods

To gain a good understanding of how memorisation is task-dependent, we combine binary classification tasks from (Super)GLUE (Wang et al., 2018, 2019) with tasks from more diverse domains and label set sizes. The tasks fall into four categories: generic *natural language understanding* (NLU), sentiment-related tasks, hate speech detection and topic classification. Table 1 enumerates the tasks, the datasets' domains, and the training set and label set sizes. For each dataset, we perturb the labels of 15% of the training examples ('noisy' examples, $x \in \mathcal{X}_n$, $y \in \mathcal{Y}_n$), with the new label randomly drawn from all labels but the original one. The remaining 85% is unperturbed ('clean' examples, $x \in \mathcal{X}_c$, $y \in \mathcal{Y}_c$).

We analyse four PLMs: `BERT-base` (Devlin et al., 2019), `OPT-125m` (Zhang et al., 2022), `Pythia-160m` (Biderman et al., 2023) and `GPT-Neo-125m` (Black et al., 2021; Gao et al., 2020). We fine-tune each model separately for the 12 tasks. Appendix D describes the models and our technical setup.[4] These 4 architectures are similar in size and have 12 layers each. In Appendix B, we repeat a subset of the experiments with `OPT-1.3B`.

The PLMs (θ_P) are fine-tuned for 50 epochs, and checkpoints are stored when the training accuracy is near-ceiling (θ_{M_1}), and at the end of training (θ_{M_2}). We also train models using the original labels (θ_O), using the same random seeds as θ_{M_1} and θ_{M_2}. During fine-tuning, we freeze the input embeddings. Results reported in §3.2 are based on one fine-tuning seed, and the remainder of the main paper computes results using three seeds. Seeds control the data order and classification heads.

3.1 Localisation techniques

We apply four localisation methods that are detailed in this subsection and further evaluated in §3.2.

Layer retraining and layer swapping First, we perform layer retraining, similar to Maini et al. (2023). We reset layers of interest using weights from θ_P, freeze the remaining layers using weights from θ_{M_2}, and retrain using clean examples for five epochs. If the resulting model maintains its performance on noisy data, the retrained layers are redundant in terms of memorisation. If the performance on noisy data decreases, that does not guarantee that memorisation can be localised to the retrained layers since the retraining objective may have multiple minima, of which only some maintain the memorisation performance. We retrain consecutive layers of window sizes ranging from 1 to 12.

[3] For instance, De Cao et al. investigate 6-layer transformers, whereas Dai et al. and Stoehr et al. use 12-layered networks, and Meng et al. mostly focus on `GPT-2 XL` with 48 layers.

[4] The codebase is available at: https://github.com/vernadankers/memorisation_localisation.

Category	Dataset	Task	Domain	Size	Labels
NLU	WiC by Pilehvar and Camacho-Collados (2019)	word sense disambiguation	Word-/Verb-net, Wiktionary	5.4k	2
	RTE by Dagan et al. (2006)	textual entailment	news, Wikipedia	2.5k	2
	MRPC by Dolan and Brockett (2005)	paraphrase classification	news	3.7k	2
	CoLA by Warstadt et al. (2019)	labelling grammatically	linguistic theory books	8.5k	2
	BoolQ by Clark et al. (2019)	question answering given a context	Google queries, Wikipedia	9.4k	2
Sentiment	SST-2 by Socher et al. (2013)	sentiment classification	movie reviews	6.9k	2
	SST-5 by Socher et al. (2013)	sentiment classification	movie reviews	8.5k	5
	Emotion by Saravia et al. (2018)	emotion classification	tweets	16k	6
Hate speech	ImplicitHate by ElSherief et al. (2021)	hate speech classification	tweets	5.1k	7
	Stormfront by de Gibert et al. (2018)	hate speech classification	social media	8.6k	2
Topic Classification	Reuters by Apte et al. (1994)	topic classification	news	5k	8
	TREC by Li and Roth (2002); Hovy et al. (2001)	topic classification	news, misc.	5.5k	6

Table 1: Datasets with their domain, label set size and training set size. In §4 and §5, datasets are marked consistently using the same colours and symbols.

Alternatively, we swap layers between θ_{M_2} and θ_O, using the same window sizes. If swapping layers leads to a drop in performance on noisy examples while maintaining performance on clean ones, it becomes more likely that the layers were vital for memorisation (although this is again not guaranteed). We indicate layer relevance using the **memorisation error**: the ratio of incorrect predictions for noisy examples. The lower the error rate for noisy examples when retraining or swapping a layer, the less likely it is that this layer was crucial for memorisation.

Retraining or swapping all 12 layers means modifying the full model, and provides a baseline for the maximum error we can expect for the noisy data. In the results section, we will use this to normalise the results, such that the memorisation error is 1.0 when modifying all 12 layers.

Forgetting gradients We also inspect gradients, computed by back-propagating $-\mathcal{L}(\mathcal{X}_n, \mathcal{Y}_n, \theta_{M_1})$ and computing the L_1-norm per layer. We use θ_{M_1} due to gradient saturation in θ_{M_2}.[5] The assumption is that memorisation is localised in the layers requiring the largest updates when 'forgetting' noisy labels. Because gradient magnitudes do not reliably pinpoint layers, we used two tasks to decide on the norm to use and whether or not to normalise gradients using gradients for clean examples and gradients for a frozen model (Appendix E).

Probing Lastly, we train probing classifiers (Conneau et al., 2018) to predict whether, for a hidden state encoding x in layer l (h_l^x), $x \in \mathcal{X}_n$ or $x \in \mathcal{X}_c$. The classifier is an MLP with one hidden layer, trained for 100 epochs maximum with a learning rate of $2e{-}4$. The hidden states come from *training* examples that are redistributed into a training, validation and test set for the probe. The classifiers are trained separately per layer, using five random seeds per layer. We extract the F_1-score on the test partitions and use the increase from $l{-}1$ to l as an indication of l's involvement in memorisation (except for layer 1, which we compare to the F_1-score from a probe trained on θ_P).

[5] See Akyürek et al. (2022) for a discussion of issues with gradient-based methods when tracing knowledge in a model.

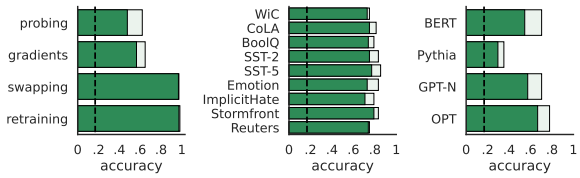
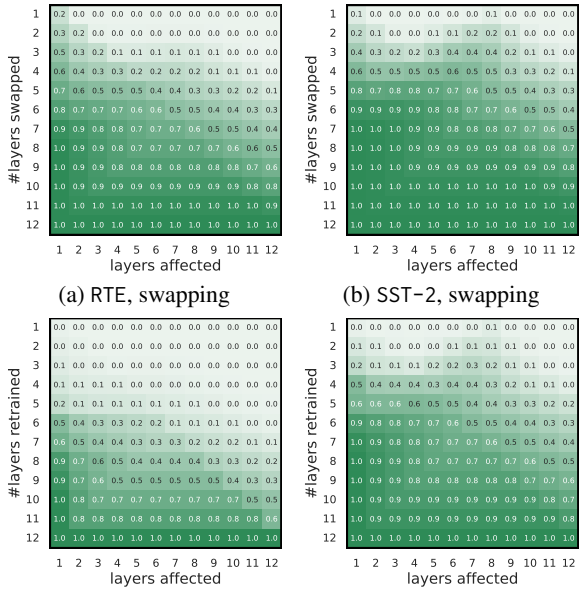


Figure 2: Control setup accuracy@1 (light) and accuracy@2 (dark) per localisation method (left), dataset (middle) or model (computed using probing and gradients, right), and a random guessing baseline (dashed).

3.2 Control setup: does localisation succeed?

We now evaluate the localisation techniques by enforcing memorisation in prespecified layers and examining whether the techniques pinpoint those layers (i.e. whether localisation succeeds).

Experimental setup We approach this as a multitask learning setup, to ensure all layers are fine-tuned, but only two are modified by the task with noisy labels: the entire model is fine-tuned using RTE, while the remaining task can only modify two layers at a time (layers 1 and 2, 6 and 7 or 11 and 12). We train the model separately for the remaining 11 tasks and these 3 different choices of layer combinations. Afterwards, we first use MRPC and TREC to validate the postprocessing steps for the forgetting gradients (see Appendix E), after which all localisation techniques were applied to the remaining nine tasks. We evaluate the techniques using **accuracy@k**, indicating the percentage of the k highest-scoring layers that were among the correct ones for that setup, computed for $k \in \{1, 2\}$.



(a) RTE, swapping (b) SST-2, swapping

(c) RTE, retraining (d) SST-2, retraining

Figure 3: Memorisation error for layer swapping and retraining for two datasets, for the OPT model.

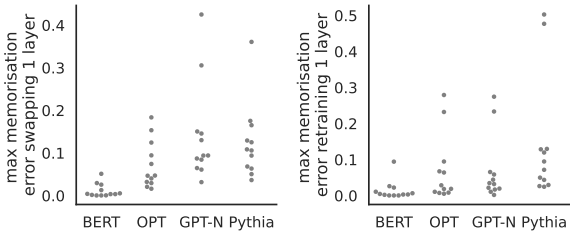
Results Figure 2 (left) summarises the accuracies per localisation technique. Swapping and retraining are very accurate, but gradients and probing are less reliable, with accuracy@1 just over 60%. Note that the near-perfect accuracy for retraining and swapping here does not guarantee perfect accuracy in the uncontrolled setup; the per-layer freezing is just very well-aligned with the per-layer approach of those techniques. The accuracy per dataset (Figure 2, middle) only shows slight variations. For the two lowest-scoring localisation techniques (probing, gradients), Figure 2 (right) details the accuracies per model. Pythia scores particularly badly for the gradient analysis, for which the accuracies barely exceed the baseline. Postprocessing (Appendix E) did not help, which underscores gradients' unreliability.

4 Results for memorisation localisation

We now apply the localisation techniques to models in which all layers have been fine-tuned for one task at a time. The results indicate how important each layer is for memorisation, per dataset, per model. We cannot simply aggregate over all results (12 layers × 12 datasets × 4 localisation techniques × 4 models), because the absolute scores returned by different techniques are not directly comparable. We discuss the results per localisation technique.

4.1 Layer swapping and retraining

When swapping or retraining layers, we gradually modify more and more layers in θ_{M_2}, either using weights from θ_O, or by retraining layers using clean examples.[6] We modify 1 to 12 layers at a time, and measure the effect via the memorisation error.



(a) Layer swapping (b) Layer retraining

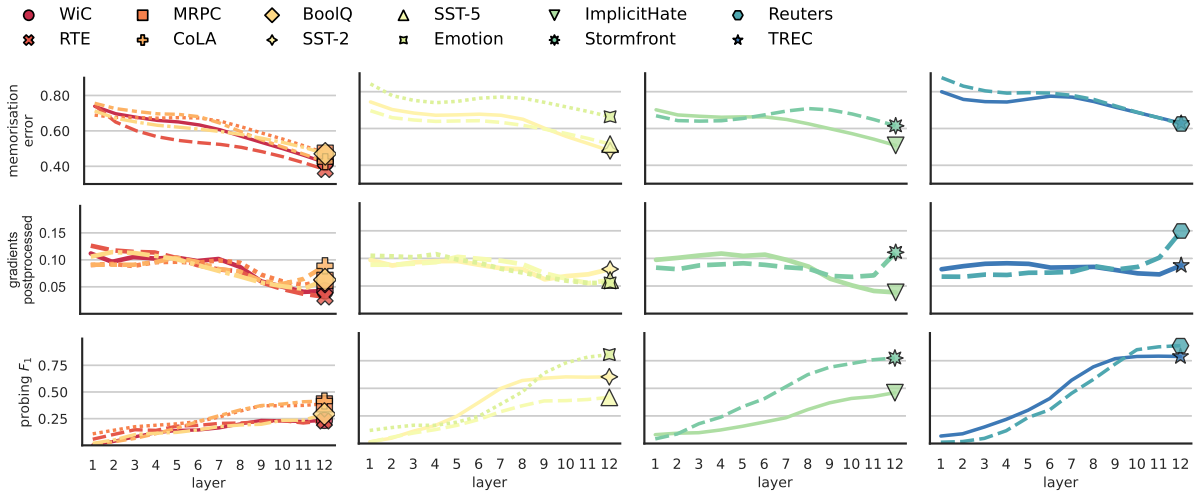
Figure 4: Maximum memorisation error over 12 layers when modifying 1 layer; dots represent datasets. Jitter along the x-axis was added to improve visibility.

Case study: RTE vs SST-2 Before discussing trends across all 12 datasets, we inspect two specific sets of results to gain a deeper understanding of the data. Figure 3 details memorisation error rates for RTE and SST-2 (for OPT): in these matrices, value z in row x, column y, indicates that for all layer combinations of x consecutive layers including y, z was the mean error rate. We show the results separately for swapping and retraining.

What commonalities and differences do we observe? For both datasets, modifying a few layers only yields low error rates (see the top few light green rows), and fully reverting memorisation requires modifying 7 to 10 layers. Memorisation is thus not limited to a few layers, but, instead, dispersed over the model. Despite these similarities, the datasets differ in which layers appear the most crucial for memorisation: for RTE, modifying early layers leads to the largest increase in memorisation error, whereas for SST-2, both the very first layers and layers in the middle appear most relevant.

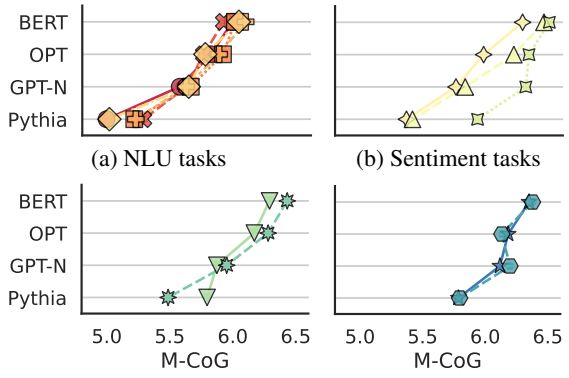
Aggregating results The findings for these two tasks are echoed in the overall swapping and retraining results. Firstly, **memorisation is not confined to individual layers**: modifying individual layers barely affects the memorisation error. This is shown in Figure 4, which provides the memorisation error when modifying one layer only, taking the *maximum* over layers (i.e. highlighting the largest error increase), showing datasets as dots. For most model-dataset combinations, the memorisation error rate is below 15% when modifying one layer. This agrees with findings from Maini et al. (2023), who similarly employed layer retraining to identify that memorisation in image classification

[6] When swapping layers, we monitor errors on clean examples to ensure that the mixture of models θ_O and θ_{M_2} differs only in terms of predictions for noisy examples. The mean error for clean examples over all windows was 0.3%.



(a) NLU tasks (b) Sentiment tasks (c) Hate speech tasks (d) Topic classification

Figure 5: Memorisation localisation for OPT: (1) layer swapping error rates, higher numbers suggest higher relevance. (2) gradient norms, higher numbers suggest higher relevance. (3) probing F_1-scores when training probes to predict whether an example is noisy. The increase between layers suggests layer relevance.



(a) NLU tasks (b) Sentiment tasks

(c) Hate speech tasks (d) Topic classification

Figure 6: M-CoG coefficients for layer retraining, that give a coarse indication of whether lower or higher layers matter more for memorisation.

is not confined to individual layers.

Secondly, **the importance of layers does appear task-dependent**. To investigate this more systematically, we express layer relevance using the mean memorisation error (averaging over rows in the result matrices similar to the ones in Figure 3). Figure 5 (top row demonstrating layer swapping) details this error per dataset for OPT. Across the board, **early layers matter more for memorisation**, but that is more prominent for the NLU tasks than for the other tasks, and for Stormfront there is even a slight increase in the error for *deeper* layers. For the remaining models, the same visualisation is shown in Appendix A.1, and we can summarise the per-layer weights by computing a *Memorisation Centre-of-Gravity* (M-CoG), which is a weighted sum of all layers with weights summing to 1: $\sum_{i=1}^{12} \alpha_i \cdot i$. For layer swapping and retraining, α_i is the memorisation error for layer i, as depicted in Figure 5. Figure 6 displays the M-CoG coefficients for layer retraining, per model, and Figure 7 provides M-CoG coefficients per dataset by averaging over models (left) and over localisation techniques (right). The results show **strong agreement between models in terms of the relative ordering of tasks** – the average pairwise correlation of the data from Figure 6 is 0.85 (Spearman's ρ) – and between layer swapping and layer retraining – Figure 8 (left) includes rank correlations for the M-CoG coefficients, and Figure 9 (left) includes rank correlations for raw layer weights. Both indicate strong agreement between the two techniques.

4.2 Probing

Figure 5 (bottom row) displays the probing performance for OPT, and the increase from layer to layer indicates layers' relevance. The first observation is that the performance typically does not decrease for deeper layers, i.e. representations do not 'lose' information about the fact that some examples are noisy. Secondly, the performance is quite low for NLU tasks, especially, which could mean that clean and noisy examples are more alike for these tasks than for the remaining tasks. Lastly, in accordance with the previous results, probing performance does not change suddenly (i.e. **memorisation is not local to individual layers**), and **tasks differ in how the probing performance changes over layers**: performance flattens early for some tasks (e.g. RTE) but gradually improves over all layers for others (e.g. Emotion, ImplicitHate). Appendix A.1 provides results for the other models; for Pythia, probing performance peaks earlier

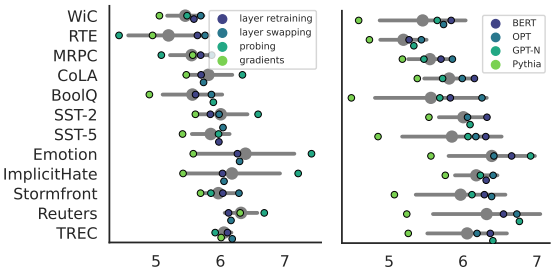


Figure 7: M-CoG coefficients averaged over models (left) and averaged over localisation techniques (right). Error bars show standard deviations.

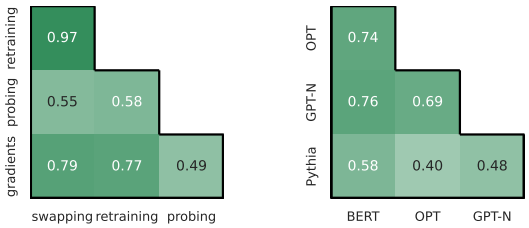


Figure 8: Spearman's ρ for M-CoG from different localisation techniques (left), and different models (right).

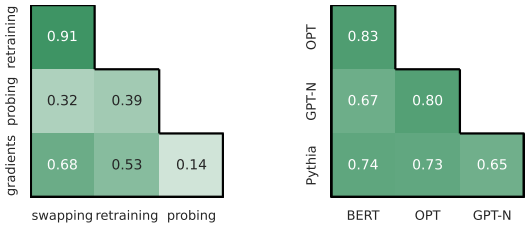


Figure 9: Spearman's ρ for layer-wise scores from different localisation techniques (left), and models (right). When comparing models, we collect weights from 4 techniques. Those are not directly comparable, so we apply min-max normalisation per technique.

than for the other models, indicating that the lower layers are extra important for this model.

To draw more generic conclusions, we compute the M-CoG coefficients by using the per-layer increase in probing performance as weights. Figure 7 (left) includes the M-CoG averaged over models and demonstrates that probing puts a larger emphasis on deeper layers compared to layer swapping and retraining. The M-CoG of probing have a moderately positive correlation to the swapping and retraining coefficients (see Figure 8), and raw weights per layer have a weakly positive correlation to swapping and retraining (see Figure 9).

4.3 Gradient analysis

Finally, we inspect the gradient norms, post-processed as described in Appendix E. Results obtained using the forgetting gradients correlate quite strongly with layer swapping and retraining (Figure 8, Figure 9). That agreement can also be seen when visually inspecting the norms per layer for `OPT` (middle row of Figure 5, see Appendix A.1 for the remaining models): NLU tasks have higher scores in earlier layers, `SST-2` and `TREC` have a more uniform distribution, and `Stormfront` and `Reuters` point to deeper layers (although the gradient norms show a slight increase for the final layer for multiple tasks). At the same time, the gradient analysis weakly correlates to the probing results, potentially because *both* methods have much lower accuracies than swapping/retraining. The forgetting gradients failed to pinpoint one model's correct layers in the control setup (§3.2). *That* gradients agree with swapping/retraining supports our overall findings, but we recommend against relying solely on gradients for memorisation localisation.

4.4 Intermediate conclusion

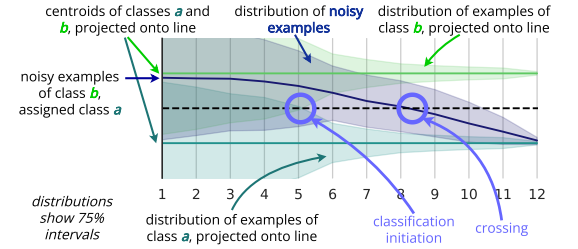
In this section, we have taken a closer look at the localisation results for `OPT`, and inspected aggregated results for all techniques and models via M-CoG coefficients. Because memorisation is not strictly localised to individual layers, these coefficients lie close to the middle layer, but they do generally skew towards earlier layers and provide us with an ordering of tasks. The most notable pattern in that ordering that the earlier layers are the most important for the NLU tasks. This is somewhat surprising since the NLP community would typically consider NLU tasks to be more complex than topic classification or sentiment detection, and assumes higher-level tasks to be processed in higher layers.[7] If that is the case, it seems natural for memorisation to also happen in higher layers, but this is contradicted by the experiments.

While this section has concentrated primarily on the comparison of localisation methods, we finally note that when computing correlations between models (Figure 8-9, right), these are strongly positive, except for `Pythia`, yielding more moderate correlations. That suggests that our results are not specific to one training setup, but somewhat generic to 12-layer transformer-based PLMs.

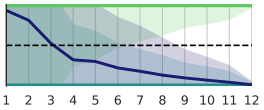
5 Making memorisation interpretable via centroid analysis and probing

The results from §4 suggested that earlier layers are the most relevant for memorisation. To better understand why, we make models' internal processing of memorised examples more interpretable through a **centroid analysis**: we examine pairs of

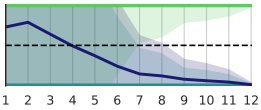
[7] E.g. Müller-Eberstein et al. (2023) show that for topic classification in `BERT` (using unperturbed datasets), the centre-of-gravity as defined by Tenney et al. (2019) lies around layer 4/5 for topic classification, whereas for NLI it is layer 11.



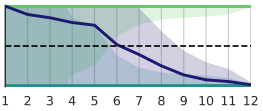
(a) TREC



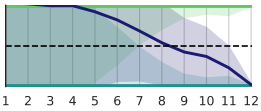
(b) RTE



(c) WiC

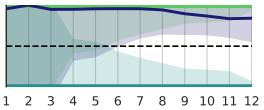


(d) SST-2

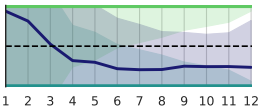


(e) Stormfront

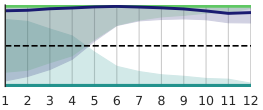
Figure 10: Centroid analysis visualises how noisy examples gradually change, for five datasets, for OPT.



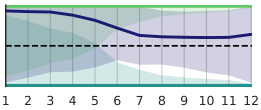
(a) RTE, regular bottom



(b) RTE, regular top



(c) TREC, regular bottom



(d) TREC, regular top

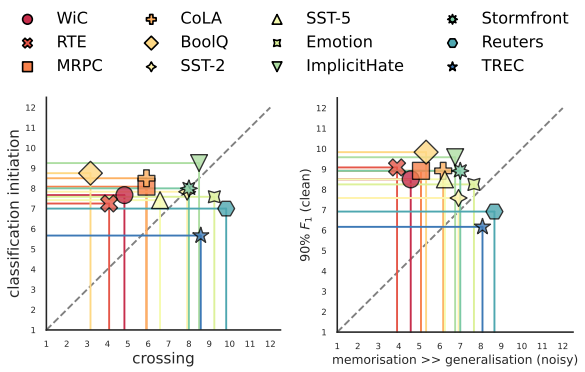
Figure 11: Illustration of the effect replacing 6 layers has, for RTE and TREC data, for OPT. We insert 6 layers from θ_O at the lower ('bottom') or upper half ('top').

classes, monitoring examples with original class y_b and noisy class y_a, for all pairs of a and b. We compute the centroids of the hidden representations from the two classes and project all data points from those two classes onto the line through the centroids. We measure the distance to centroid a for every data point, normalised by the distance between the two centroids. This is performed separately per layer. In layer 1, points belonging to y_a and y_b largely overlap. Towards layer 12, the two classes are fully separated, and in between, the memorised examples move away from centroid b and move towards centroid a. Figure 10a explains this via annotations for TREC, and Figures 10b-10e do so for four additional tasks that are illustrative of the variety that we observe. We include figures for all tasks and models in Appendix A.3.

This visualisation indicates that memorisation occurs through *gradual* changes from the first layers onward. This explains the results from the previous section, where we found that memorisation is not confined to individual layers and that lower layers were more successful in reverting memorisation (in layer swapping/retraining) than deeper layers: memorisation *starts* early, and interventions are more successful when conducted before the hidden state has moved too far away from class y_b. We can demonstrate this using the centroid analysis, applied while swapping six layers at the bottom or top with layers from θ_O. For all tasks (see Appendix A.3), we can prevent the noisy examples from moving to centroid a by replacing the bottom six, but for only a few tasks, replacing the top six has a similar effect. Figure 11 shows this using RTE and TREC. Swapping the bottom six prevents emitting the noisy class for RTE *and* TREC, but swapping the top six is more successful for TREC than RTE.

Task differences To summarise task differences, we compute two statistics: the **crossing** (the first layer in which the noisy mean is closer to a than to b) and **classification initiation** (the first layer without overlapping distributions for the two classes), shown in Figure 12a. Many NLU tasks have an early crossing and a late classification initiation (e.g. RTE and WiC). The two events are closer together for hate speech and sentiment tasks, and topic classification tasks (TREC and Reuters) start classification early but have a late crossing. This confirms findings from §4: lower-level tasks (early classification initiation) rely more heavily on deeper layers for memorisation than higher-level tasks (late classification initiation). Yet, tasks with similar classification initiations (e.g. SST-5 and Emotion) can still have different crossings.

Consolidation via probing The centroid analysis merely visualises representations. To consolidate that we reach similar conclusions using different methods, we train probes to predict an example's class from the hidden state, using (i) original or (ii) noisy labels. In Appendix A.2, we include the probes' performances. We apply the probes to noisy examples and compute a statistic similar to the crossing: the layer at which the F_1 of probe ii exceeds the F_1 of probe i with ten percentage points, referred to as 'memorisation»generalisation' in Figure 12b. The timing of this event strongly correlates with the crossings (Spearman's $\rho = 0.84$). We apply the probes to clean examples to compute a statistic similar to the classification initiation: the layer at which the probes' F_1 for clean examples (normalised by random guessing performance) reaches 90%. The depth of this event strongly correlates with classification initiation ($\rho = 0.73$). Together, these two events thus tell a story similar to that of the centroid analysis (Figure 12b).



(a) Centroid analysis (b) Probing

Figure 12: Summary of the memorisation and classification onset for all datasets, averaged over models, computed using the centroid analysis or via probing.

Correlates	all	BERT	OPT	GPT-N	Pythia
- *Generalisation score*					
crossing	0.75	0.88	0.94	0.94	0.72
mem. » gen.	0.63	0.86	0.88	0.94	0.69
M-CoG	0.56	0.78	0.69	0.92	0.69
- *Validation score*					
crossing	0.70	0.90	0.84	0.83	0.70
mem. » gen.	0.61	0.90	0.77	0.76	0.77
M-CoG	0.54	0.80	0.52*	0.72	0.69

Table 2: Spearman's ρ relating memorisation for the 12 tasks to models' generalisation performances. *: p>0.05

Memorisation's connection to generalisation When inspecting model internals, we have seen that the depth of memorisation (quantified as M-CoG coefficients, the 'crossing' and 'memorisation»generalisation') appears anti-correlated with the difficulty of a task. However, we have yet to have a proper way of quantifying that difficulty. We now take models' **validation accuracy** at the end of training (on data unseen during training, normalised by random guessing performance) and compute a **generalisation score** (percentage of *training examples* for which the probability of the target exceeds random guessing when that example is held out from training).[8] As indicated in Table 2, these two metrics correlate moderately with the memorisation depth when combining data from all models (Spearman's $\rho > 0.54$), with most correlations being stronger when examining results per model. All in all, this suggests that the better a model generalises a task to new data, the more later layers are involved in memorisation.[9]

[8]Computed by training on a randomly selected 50% of the data, and testing on the held-out 50%, repeated with 30 random seeds. This approximates metrics reported by Feldman (2020); Feldman and Zhang (2020); Jiang et al. (2021). We adopt the naming of the metric from Dankers et al. (2023).

[9]Tasks with late crossings are mostly multi-class tasks. To ensure that this is not a confound here, Appendix C repeats some of the analyses with binary versions of these tasks.

6 Discussion

We set out to perform memorisation localisation for natural language classification tasks by perturbing a subset of the labels and tracing those 'noisy' examples over layers. Applying four localisation techniques to four models crystallised that memorisation is not local to specific layers but a cooperative process of weights from many layers. Nonetheless, not all layers appear equally important. Overall, early layers are more important than later ones: the model's manipulation of memorised examples *starts* in lower layers, and to prevent memorisation, early intervention is more successful than late intervention. We discussed results for 12-layer models in the main paper, but further experimentation with a 24-layer model (in Appendix B) leads to similar findings of memorisation being gradual, with a similar ordering of tasks, but *mid-range* layers being more important than late layers.

This is not in accordance with the generalisation-first, memorisation-second hypothesis from CV (see §2), but does agree with more recent work on image classification by Maini et al. (2023). It also aligns with related work on PLMs for fact memorisation and verbatim memorisation pointing to the lower layers (Geva et al., 2023; Stoehr et al., 2024), while also describing cooperative roles for earlier and deeper layers (Haviv et al., 2023).

The fact that memorisation is not local implies that editing model weights locally does not necessarily erase memorised information, even if a flipped label suggests this at the level of the output layer. This might be harmless when editing facts about named entities like cities (e.g. Meng et al., 2022a), yet is more worrisome when regarding memorisation of personal information (e.g. Carlini et al., 2021), and may be a reason why safety measures can easily be reversed in PLMs modified to reduce harmful outputs (e.g. Zhan et al., 2023).

Can we, due to the importance of early layers, conclude that our results falsify the generalisation first, memorisation second hypothesis? The results from §5 suggest that this question requires a nuanced answer due to the variation observed among tasks. The depth of memorisation is positively correlated with a model's generalisation capabilities, i.e. we do observe a generalisation first, memorisation second tendency but the better the model performs at a certain task, the stronger that tendency is. We consider better understanding what properties of a task direct memorisation to lower or higher layers an exciting avenue for future work.

Limitations

We identify four main limitations of our work:

- **Simplified data**: To trace memorised examples over the transformer's many layers, we resorted to label flipping to create 'noisy' examples. This situation is somewhat unnatural when considering real-world examples that require memorisation from the model. For example, in the case of sentiment analysis, that might be a sarcastic phrase whose sentiment is the opposite of what is expected based on a literal interpretation. We cannot guarantee that our noisy examples behave in the same way as real-world examples would. Similarly, memorisation of noisy examples need not affect models in the same way as the memorisation of factual information or verbatim memorisation of long strings. As laid out in the introduction (§1), we opted for this type of data manipulation to create an experimental setup that more closely resembles that of related work from CV.

- **Localisation techniques are imperfect:** As identified in §3.2 the localisation techniques applied are themselves flawed: in a control setup where only two layers were modified during fine-tuning, probing and gradient analyses could not accurately pinpoint those two layers, and the techniques that could pinpoint them (layer swapping and layer retraining) are more reliable at determining which layers are *not* crucial for memorisation than at determining which ones are. Because of the general agreement between the techniques and the results from §4-5 we do think our conclusions are robust, but the absolute numbers of layer relevance should be taken with a grain of salt.

- **Visualisation ≠ localisation:** In §5 we introduced the centroid analysis as a way of visualising what is happening to examples over the different layers. This visualisation is a one-dimensional projection of hidden representations and thus an extreme simplification of the intricate process of memorisation. We do not mean to use it as a localisation technique, but as a way to explain the outcomes of other experiments in the paper.

- **Lack of evidence for individual examples:** We analysed the group of noisy examples as a whole, and concluded that many layers work together to gradually shift examples from their original class to the newly assigned class. However, we have not examined individual examples; it can still be the case that for individual examples, memorisation is more localised to specific layers. We only have preliminary results suggesting that individual examples, too, are memorised over multiple layers, which is the fact that in §4, swapping/retraining individual layers was unsuccessful in increasing the memorisation error rate.

Acknowledgements

VD is supported by the UKRI Centre for Doctoral Training in Natural Language Processing, funded by the UKRI (grant EP/S022481/1) and the University of Edinburgh, School of Informatics and School of Philosophy, Psychology & Language Sciences. IT is supported by the Dutch National Science Foundation (NWO Vici VI.C.212.053).

References

Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. 2022. Towards tracing knowledge in language models back to the training data. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2429–2446.

Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. 2019. Intrinsic dimension of data representations in deep neural networks. *Advances in Neural Information Processing Systems*, 32:6111–6122.

Chidanand Apte, Fred Damerau, and Sholom M Weiss. 1994. Towards language independent automated learning of text categorization models. In *SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University*, pages 23–30. Springer.

Robert Baldock, Hartmut Maennel, and Behnam Neyshabur. 2021. Deep learning through the lens of example difficulty. *Advances in Neural Information Processing Systems*, 34:10876–10889.

Stella Biderman, USVSN Sai Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. 2024. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems*, 36:28072–28090.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. In *International Conference on Learning Representations*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Kent Chang, Mackenzie Cramer, Sandeep Soni, and David Bamman. 2023a. Speak, memory: An archaeology of books known to ChatGPT/GPT-4. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7312–7327.

Ting-Yun Chang, Jesse Thomason, and Robin Jia. 2023b. Do localization methods actually localize memorized data in LLMs? *arXiv preprint arXiv:2311.09060*.

Yuheng Chen, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Journey to the center of the knowledge neurons: Discoveries of language-independent knowledge neurons and degenerate knowledge neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17817–17825.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936.

Gilad Cohen, Guillermo Sapiro, and Raja Giryes. 2018. DNN or k-NN: That is the generalize vs. memorize question. *arXiv preprint arXiv:1805.06822*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*. Springer.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502.

Verna Dankers, Ivan Titov, and Dieuwke Hupkes. 2023. Memorisation cartography: Mapping out the memorisation-generalisation continuum in neural machine translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8323–8343.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506.

Ona de Gibert, Naiara Pérez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363.

Vitaly Feldman. 2020. Does learning require memorization? A short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33:2881–2891.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020.

The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235.

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

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2024. Does localization inform editing? Surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36:17643–17668.

Adi Haviv, Ido Cohen, Jacob Gidron, Roei Schuster, Yoav Goldberg, and Mor Geva. 2023. Understanding transformer memorization recall through idioms. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 248–264.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. 2021. Characterizing structural regularities of labeled data in overparameterized models. In *International Conference on Machine Learning*, pages 5034–5044. PMLR.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Pratyush Maini, Michael Curtis Mozer, Hanie Sedghi, Zachary Chase Lipton, J Zico Kolter, and Chiyuan Zhang. 2023. Can neural network memorization be localized? In *International Conference on Machine Learning*, pages 23536–23557. PMLR.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. In *International Conference on Learning Representations*.

Fatemehsadat Mireshghallah, Archit Uniyal, Tianhao Wang, David K Evans, and Taylor Berg-Kirkpatrick. 2022. An empirical analysis of memorization in fine-tuned autoregressive language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1816–1826.

Ari Morcos, Maithra Raghu, and Samy Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. *Advances in neural information processing systems*, 31.

Max Müller-Eberstein, Rob Van Der Goot, Barbara Plank, and Ivan Titov. 2023. Subspace chronicles: How linguistic information emerges, shifts and interacts during language model training. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13190–13208.

Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A Feder Cooper, Daphne Ippolito, Christopher A Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. 2023. Scalable extraction of training data from (production) language models. *arXiv preprint arXiv:2311.17035*.

Jingcheng Niu, Andrew Liu, Zining Zhu, and Gerald Penn. 2024. What does the knowledge neuron thesis have to do with knowledge? In *International Conference on Learning Representations*.

Francesco Ortu, Zhijing Jin, Diego Doimo, Mrinmaya Sachan, Alberto Cazzaniga, and Bernhard Schölkopf. 2024. Competition of mechanisms: Tracing how language models handle facts and counterfactuals. *arXiv preprint arXiv:2402.11655*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697.

Arnab Sen Sharma, David Atkinson, and David Bau. 2024. Locating and editing factual associations in Mamba. *arXiv preprint arXiv:2404.03646*.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023. Detecting pretraining data from large language models. In *NeurIPS 2023 Workshop on Regulatable ML*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Cory Stephenson, Suchismita Padhy, Abhinav Ganesh, Yue Hui, Hanlin Tang, and Sue Yeon Chung. 2021. On the geometry of generalization and memorization

in deep neural networks. In *International Conference on Learning Representations*.

Niklas Stoehr, Mitchell Gordon, Chiyuan Zhang, and Owen Lewis. 2024. Localizing paragraph memorization in language models. *arXiv preprint arXiv:2403.19851*.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2022. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578.

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

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Shenglai Zeng, Yaxin Li, Jie Ren, Yiding Liu, Han Xu, Pengfei He, Yue Xing, Shuaiqiang Wang, Jiliang Tang, and Dawei Yin. 2023. Exploring memorization in fine-tuned language models. *arXiv preprint arXiv:2310.06714*.

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori Hashimoto, and Daniel Kang. 2023. Removing RLHF protections in GPT-4 via fine-tuning. *arXiv preprint arXiv:2311.05553*.

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023. Counterfactual memorization in neural language models. *Advances in Neural Information Processing Systems*, 36:39321–39362.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

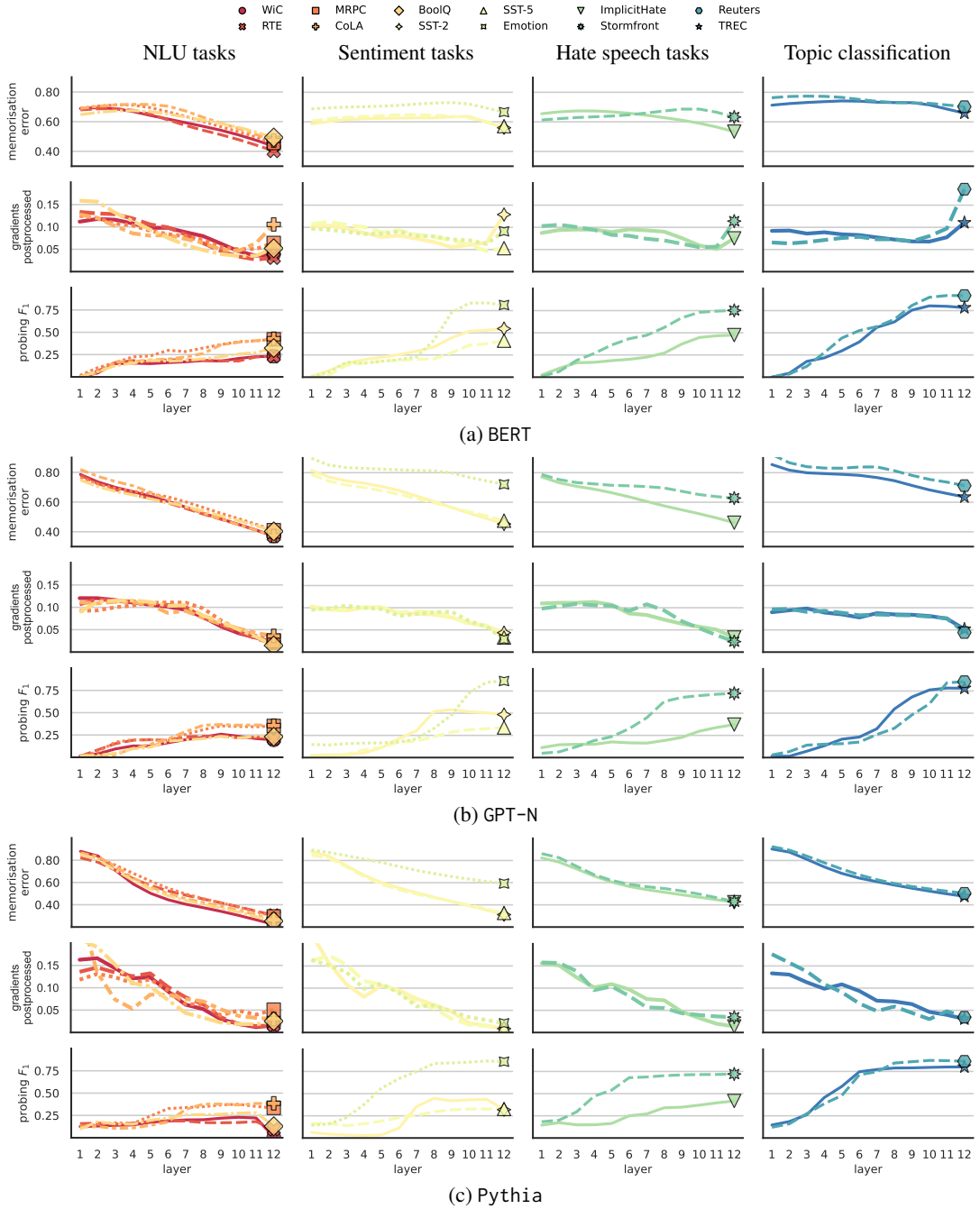
Wayne Xin Zhao, Naoki Yoshinaga, and Daisuke Oba. 2024. Tracing the roots of facts in multilingual language models: Independent, shared, and transferred knowledge. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2088–2102.

Xiaosen Zheng and Jing Jiang. 2022. An empirical study of memorization in NLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6265–6278.

A Extended results

A.1 Main results (§4)

We provide the same visualisation of results as shown for OPT in §4 in Figures 13a, for BERT, 13b, for GPT-N, and in 13c for Pythia. We omit layer retraining, that correlates very strongly with layer swapping (shown in the top rows of each subfigure).



(a) BERT

(b) GPT-N

(c) Pythia

Figure 13: Memorisation localisation techniques: (1) the top row provides the memorisation error when swapping layers (inserting non-memorisation layers into a memorisation model), higher numbers suggest higher relevance. (2) the middle row indicates (postprocessed) gradient norms, higher numbers suggest higher relevance. (3) the bottom row provides probing F_1-scores when training probes to predict whether an example is a noisy one, where the increase between layers suggests layer relevance.

A.2 Probing to consolidate centroid analysis (§5)

In §5, we trained probes to predict an example's class based on its hidden state, using either original or perturbed labels. Figure 14 shows a) test F_1-scores of the noisy examples for the original label (dashed line), b) test F_1-scores of the noisy examples for the perturbed label (solid line), and c) the performance on clean examples when training with the original labels (dotted line). Tasks vary widely in terms of when the F_1-score for noisy labels exceeds that of the original labels. This happens early on for WiC and RTE, but for other tasks (e.g. SST-2, TREC and Reuters), the probe is better at predicting the original label before it can predict the noisy one. We summarised this in the main paper using the 'memorisation » generalisation' event.

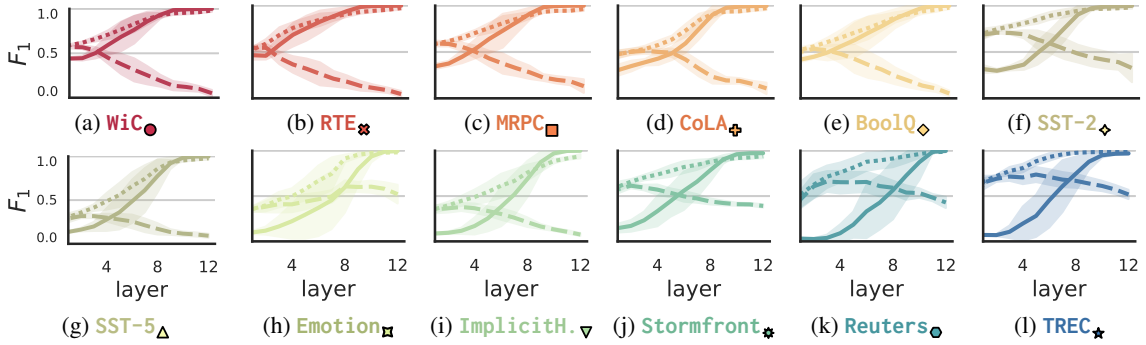


Figure 14: We train probes to predict the noisy label (solid line, shown for noisy examples) or the original label (dashed line for noisy examples, dotted line for clean examples).

A.3 Centroid analysis (§5)

In §5 we introduced the centroid analysis as a way to visualise what happens to noisy examples as they move through the different processing layers. Figure 15 now includes visualisations for all four models, and for OPT with either the bottom or the top six layers swapped with those from θ_O. Visual inspection leads to the following observations:

- **Gradual vs. relatively localised tasks:** For nearly all setups, the noisy examples gradually move from one centroid to another, confirming that memorisation is a process in which many layers are involved. Still, there are relative differences to be observed, e.g. compare TREC (gradual) to MRPC (relatively localised) for Pythia, or compare RTE (relatively localised) to ImplicitHate (gradual) for BERT.

- **Classification initiation task variation:** Tasks do not always have a consistent classification initiation across models: some tasks are relatively stable in terms of when the two distributions of a and b stop overlapping (e.g. WiC), others show great variation across the four models (e.g. Reuters, Stormfront, Emotion).

- **Swapping bottom layers most successful:** Inspecting the swapping centroid visualisations (final two columns) demonstrates that swapping out the bottom six layers of a memorisation model with θ_O can prevent the 'crossing' (see §5) from happening. Swapping out the top six layers, on the other hand, is less successful since, for some datasets, the memorised examples have already 'crossed' (e.g. see WiC, RTE and MRPC).

- **Centroid analysis is a simplification:** There are a few dataset-model combinations that show that the centroid analysis is not always a straightforward way to explain the model's behaviour. For instance, for Pythia, CoLA leads to unintuitive results, mostly due to the fact that the two centroids nearly overlap, making the relative distance between them less meaningful in the early layers.

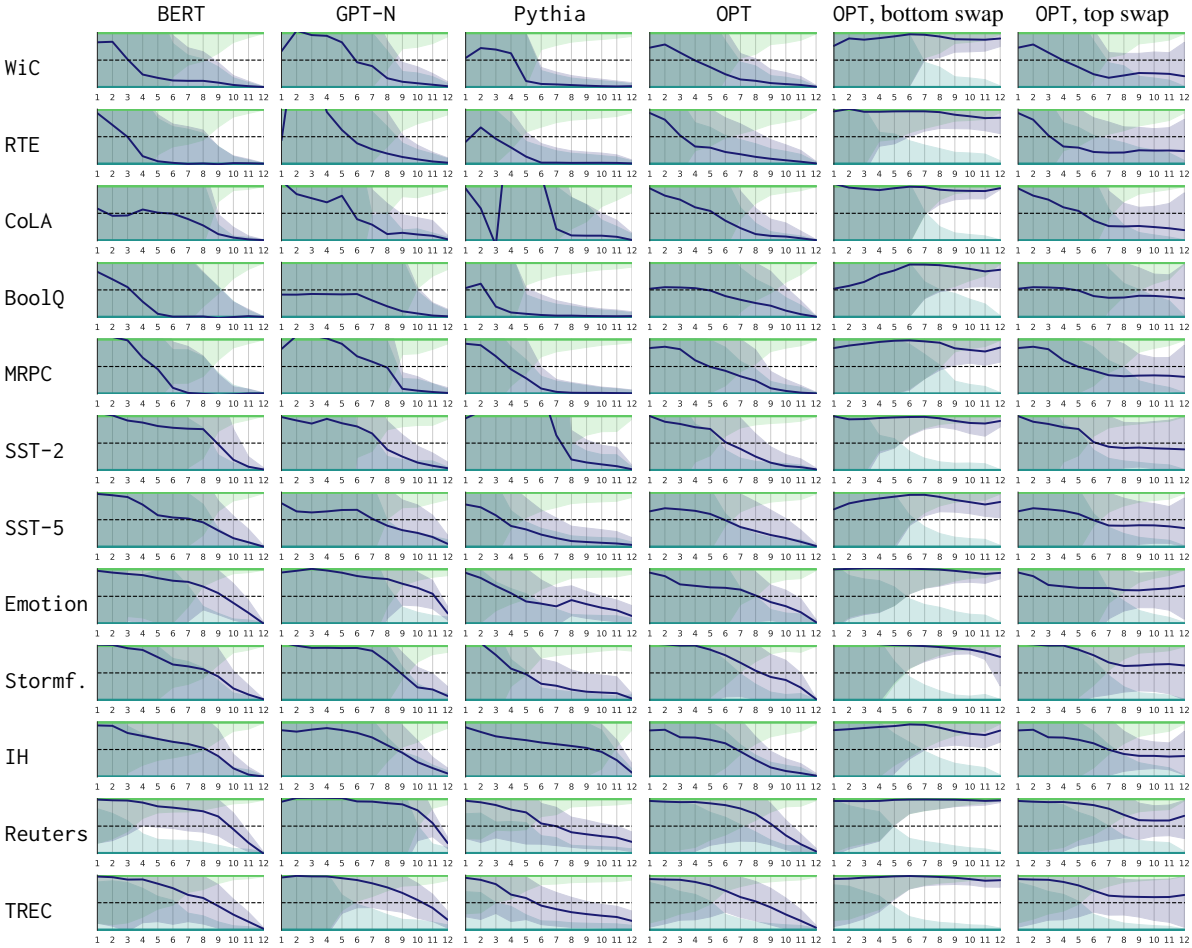


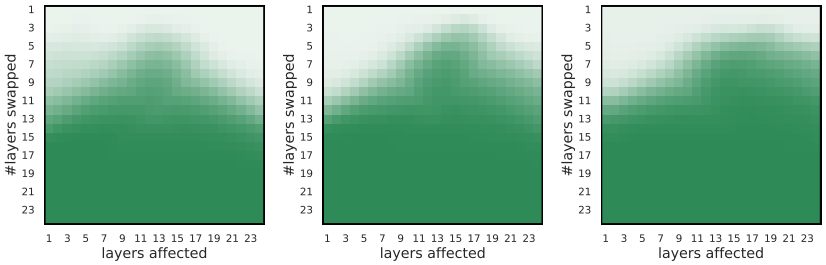
Figure 15: Visualisations of the centroid analyses for all four models, including additional visualisations for OPT where the model either is assigned a 'regular' bottom 6 layers, or a 'regular' top 6 layers (regular meaning the layers come from θ_O, trained on the original labels).

B Increasing model size

In the main paper, we have discussed results for four 12-layer architectures, observing generally strong agreement across those models, in spite of the fact that they were trained with varying corpora and for varying numbers of updates. To examine to what extent the results observed were specific to *12*-layer architectures, we apply layer swapping to the 1.3B variant of OPT, containing 24 layers and ten times the number of parameters of the other models we considered.

Figure 16 firstly provides three example matrices, similar to the ones discussed in §4.1. For all three datasets shown, swapping the middle layers most effectively reverts memorisation when considering the smaller window sizes, but there are clear distinctions between the three datasets, too: for WiC only the middle layers appear most relevant, whereas for SST-2 and TREC the upper layers are more relevant than for WiC. Figure 18 averages the rows from the matrices to summarise results across the 12 datasets, displaying a pattern similar to the main paper, with NLU tasks relying more heavily on (relatively speaking) lower layers than the remaining tasks. The agreement is also reflected in Spearman's ρ between the M-CoG coefficients from the main paper for layer swapping and the M-CoG coefficients computed using Figure 18: $\rho = 0.73$ for Pythia, $\rho = 0.84$ for GPT-N, $\rho = 0.75$ for BERT and $\rho = 0.87$ for OPT (small). At the same time, there is a difference to the results discussed in the main paper since the lowest layers (in absolute terms) appear much less relevant.

When we execute the centroid analysis and summarise the results using the crossing and classification initiation events (Figure 17), we similarly observe that the crossings correlate very strongly with the crossings from the four models ($\rho = 0.94$), although the classification initiations correlate very weakly with OPT-1.3B ($\rho = 0.15$, but $p > 0.05$).



(a) WiC (b) SST-2 (c) TREC

Figure 16: Layer swapping results for three datasets, for OPT-1.3B containing 24 layers. The graphs show the error rate for noisy examples, that goes from 0% when swapping only 1 layer to 100% when swapping all layers.

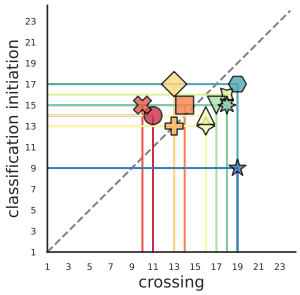
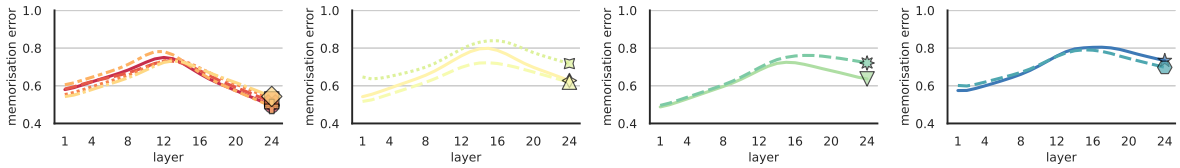


Figure 17: Summary of the memorisation and classification onsets for OPT-1.3B.



(a) NLU tasks (b) Sentiment tasks (c) Hate speech tasks (d) Topic classification

Figure 18: Per-layer memorisation error rate, averaged over all window sizes during layer swapping for OPT-1.3B. A higher error rate suggests higher relevance for memorisation.

C Binarisation of tasks

In the main paper, we used 12 varied NLP classification tasks in our memorisation localisation endeavours. Having identified that tasks differ in terms of the layers that matter most for memorisation, we should also note that the tasks with the highest M-CoG coefficients and the highest crossings in §5 also happen to be the tasks that do not have a binary label set – e.g. consider Figure 12a, where among the six highest crossings, there are five from multi-class tasks. To ensure that the effect observed is not specific to tasks with a large label set size, we now change the multi-class tasks (SST-5, Emotion, ImplicitHate, TREC, Reuters) into binary classification and repeat layer swapping and the centroid analysis. We do this by taking the most frequent two classes for a task, and training models again with 15% of the labels perturbed, using one model seed only. We now compare these models to the same model seed trained on the multi-class variant of the same tasks.

For layer swapping, the M-CoG of the multi-class and binary setups correlate with Spearman's $\rho = 0.84$, combining data points from all four models (see Figure 19); those same coefficients have a mean difference of -0.05 and a mean absolute difference of 0.16, meaning that overall, the coefficients differ only slightly.

When we repeat the centroid analysis and compute the crossing and classification initiation events, those similarly correlate strongly before and after binarisation ($\rho = 0.90$ with $p < 0.05$ for the crossing and $\rho = 0.67$ with $p > 0.05$ for the classification initiation). Figure 20 shows the events when averaged over models. And when we look at the absolute numbers obtained for these two events, the crossing is an average of 0.85 layers earlier, and the initiation is an average of 0.45 layers later, meaning that although the binarised tasks yield slightly different results, they still starkly differ from the results obtained for the group of NLU tasks.

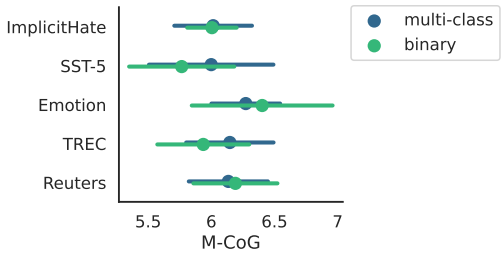
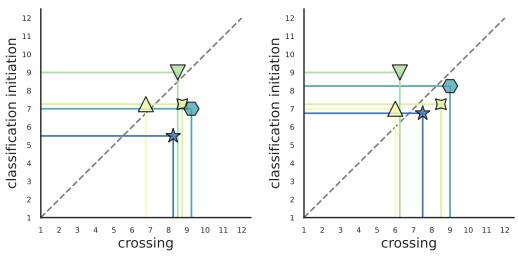


Figure 19: M-CoG coefficients for layer swapping, comparing multi-class to binarised tasks. Error bars show standard deviations over models.



(a) Multi-class tasks (b) Binarised tasks

Figure 20: Summary of the memorisation and classification onsets for the binarised multi-class tasks.

D Technical setup and model/data details

Technical setup We ran the experiments for the 12-layer models on NVIDIA GeForce RTX 1080/2080 Ti GPUs. We train the small models using a batch size of 8 (due to GPU restrictions, or 4 in the few cases where we still get memory errors, which happens for `Reuters`, in particular) and an initial learning rate of 1e-5 for 50 epochs, capping sequences at 512 tokens. 50 epochs is beyond the point of convergence since the aim is to investigate memorisation rather than optimise models for their generalisation capabilities. For the models from §3.2 where the main task can only modify two layers at a time, we rerun training with an increased learning rate if the training accuracy does not exceed .99. For every model trained, we store checkpoint θ_{M_1} when the training accuracy exceeds .993, and store checkpoint θ_{M_2} at the end of training. The most time-consuming experiments are model training and layer retraining:

- §3.2: 11 datasets × 3 control setups to obtain θ_M + 11 datasets × 3 control setups to obtain θ_O + 11 datasets × 1 frozen model = 77 setups trained for each of the 4 models, taking 1 - 6 hours each
- §4: 12 datasets × 3 seeds for θ_M + 12 datasets × 3 seeds for θ_O + 12 datasets × 1 frozen model = 84 setups trained for each of the 4 models, taking 1 - 6 hours each
  Layer retraining: 12 datasets × 3 seeds θ_M × 78 windows = 2808 setups trained for each of the 4 models, taking 3 to 45 minutes each

The experiments discussed in Appendix B are ran on NVIDIA A100-SXM80GB GPUs. `OPT-1.3B` is trained with an initial learning rate of $5e-6$ and a batch size of 32 or 16. We train two models per dataset (θ_M and θ_O), and individual training runs take 45 minutes to 6 hours, depending on the dataset. Visit our codebase here: https://github.com/vernadankers/memorisation_localisation.

We use the `transformers` library[10] to obtain the models/tokenisers and train them, implementing the remaining analyses ourselves.

Model licenses The licenses of all models, which are `Apache 2.0` (`BERT`), a custom license for `OPT` models[11] and the MIT License (`Pythia`, `GPT-N`) allow non-commercial use for research purposes.

Dataset licenses The datasets contained in GLUE and SuperGlue are available under licences that allow use and redistribution for research purposes (Wang et al., 2018, 2019). `Stormfront` is available under `CC-by-SA-3.0`; `ImplicitHate` is not explicitly assigned a license, but the corresponding repository is available under the `MIT` license; `Reuters` is available under the `CC-BY-4.0` license; for `TREC` the license is unknown, and `Emotion` should be used for educational and research purposes only, and has no license, otherwise[12].

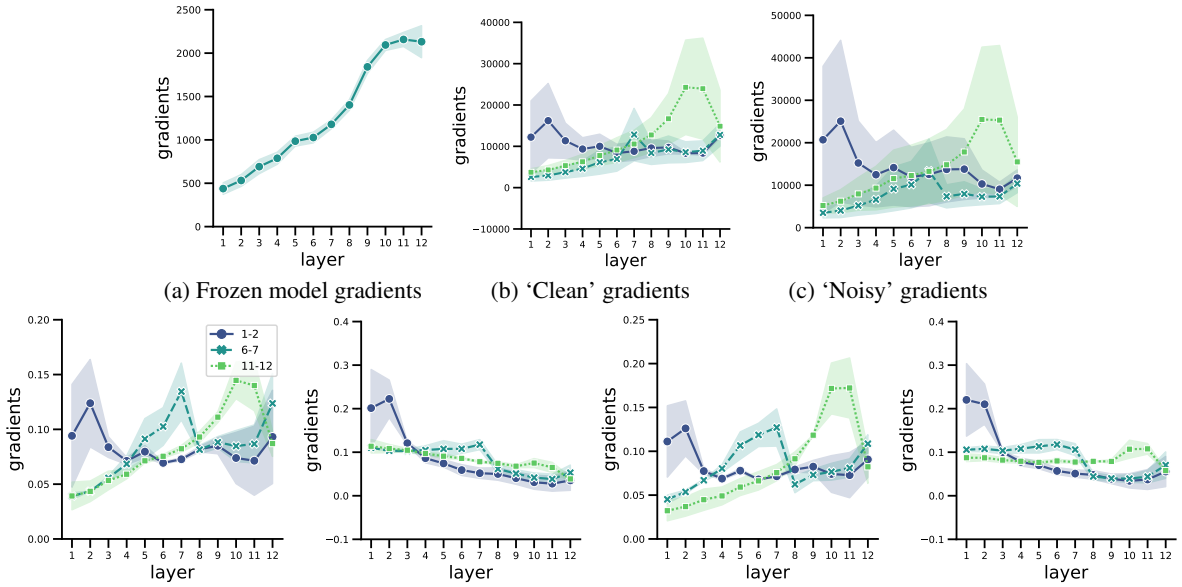
Model	Corpus	Tokens	Steps	Params	Layers	Model dim
`BERT-base`	BooksCorpus, Wikipedia	3.3B	1M	85M	12	768
`Pythia-160m`	The Pile	300B	143k	85M	12	768
`GPT-Neo-125m`	The Pile	300B	572k	85M	12	768
`OPT-125m`	BookCorpus, CC-Stories, The Pile, Reddit, CCNewsV2	180B	?	85M	12	768
`OPT-1.3B`	idem	180B	?	1.2B	24	2048

Table 3: Overview of models, along with their pre-training corpora, the number of tokens the model has seen during training, the number of training steps, the number of non-embedding parameters, and the number of layers and hidden dimensionality.

E Postprocessing gradients

As described in §3, forgetting gradients are one of the signals we examine to perform memorisation localisation. We average them over all noisy examples, or over a similar amount of clean examples. Preliminary experiments indicated that, taken at face value, the gradients do not necessarily pinpoint the

[10] https://huggingface.co/docs/transformers
[11] https://github.com/facebookresearch/metaseq/blob/main/projects/OPT/MODEL_LICENSE.md
[12] https://github.com/dair-ai/emotion_dataset



(a) Frozen model gradients (b) 'Clean' gradients (c) 'Noisy' gradients

(d) 'Noisy' gradients, norm. (e) Divide by frozen, norm. (f) Subtract clean, norm. (g) Divide, subtract, norm.

Figure 21: Effect of the gradient analysis postprocessing steps on the MRPC and TREC tasks for the BERT model when using the L_1-norm.

correct layers in a control setup. Using two validation tasks (MRPC and TREC), we consider taking the L_1-norm or the L_2-norm over gradients and applying two ways of normalising the forgetting gradients of the noisy examples: i) subtract the forgetting gradients of clean examples, ii) normalise the per-layer norm by the norm obtained using a frozen model. The final post-processing step applied afterwards is that the weights of the 12 layers are normalised to sum to 1 to allow for the computation of the M-CoG coefficients, and to reduce variation among tasks.

Figure 21a illustrates the L_1-norm for 'forgetting' gradients for a frozen BERT, that tend to point to the final layers; Figure 21b and Figure 21c demonstrate forgetting gradients for clean and noisy examples in the control setup. Both point to similar layers, but the norms are higher for noisy examples.

Figure 21d-21g *do* apply the within-dataset normalisation that normalises layer weights to sum to one. Figure 21d again demonstrates for noisy examples that without further post-processing, the gradients overestimate the relevance of later layers in BERT. Both post-processing steps i) and ii) dampen that. When measuring the success of the post-processing steps using the accuracy metric, included in Table 4, the combination of both is most successful at recovering the layers in which memorisation had taken place in the control setups, and the L_1-norm leads to more accurate results than the L_2-norm.

These post-processing steps improve the accuracy for all models but Pythia. Across the board, applying both steps i) and ii) and using the L_1-norm yields the highest accuracy, so we apply both of these steps in the main paper.

		Pythia		GPT-N		BERT		OPT	
subtracing clean	**normalising frozen**	L_1	L_2	L_1	L_2	L_1	L_2	L_1	L_2
×	×	0.08	0.08	0.58	0.25	0.58	0.50	0.58	0.17
×	✓	0.08	0.08	0.67	0.42	0.50	0.50	0.50	0.42
✓	×	0.08	0.08	0.58	0.25	0.58	0.50	0.67	0.33
✓	✓	0.08	0.00	0.75	0.25	0.75	0.58	0.58	0.50

Table 4: Effect of the gradient analysis postprocessing steps on the MRPC and TREC tasks, measured as the average accuracy of the highest scoring layers.