AGING WITH GRACE: LIFELONG MODEL EDITING WITH DISCRETE KEY-VALUE ADAPTORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models often err during deployment, either due to non-representative training data or distribution shift in the test set. Recently, *model editors* have been proposed to fix errors by adjusting a pre-trained model's weights. So far, however, existing model editors fail when making sequential edits by quickly decaying a model's performance on its upstream data. Further, when editing deployed online models, they quickly forget how to fix previously-seen mistakes. We advance beyond these existing methods by proposing and studying a novel *Lifelong Model Editing* setting, where errors stream into a deployed model and we update the model to correct its predictions without influencing its predictions for unrelated inputs. Towards effective methods in this challenging setting, we propose with General Retrieval Adaptors for Continual Editing, or GRACE. GRACE is a new Key-Value framework that casts model editing as a codebook update problem. The proposed approach edits selected model layers by caching activations that are queried using embeddings from the previous layer. The cached activations are trained to correct a model's predictions, treating future layers as a decoder. As edits stream in, the keys and values of a GRACE layer are updated while the model weights remain frozen, ensuring similar edits are treated similarly without altering the model's performance on unrelated instances. Experimentally, we show that GRACE substantially improves over recent model editors.

1 INTRODUCTION

Modern machine learning systems perform extremely well on challenging, real-world tasks. Many of the success stems from giant models trained on massive amounts of data, achieving state-of-the-art performance on various challenging tasks in natural language processing (Rae et al., 2021; Brown et al., 2020) and computer vision (Ramesh et al., 2022; Dosovitskiy et al., 2020). However, despite high performance, large models still make critical mistakes during deployment (Sinitsin et al., 2019). Further, when models are deployed over long periods of time their error rates increase as data distributions shift, labels shift as annotation guidelines change, or ground-truth information about the world simply changes. For example, a language model trained in 2016 would correctly Barack Obama as president of the United States, but this would be incorrect in 2017 and 2021. As models contain more knowledge, the correct answers will change many times during deployment (Roberts et al., 2020), as illustrated in Figure 1. In such cases, *Lifelong Model Editing* methods are an ideal fix to large models' mistakes because such edits do not incur costly retraining or degrade the model's upstream performance (Mitchell et al., 2022a).

One approach to lifelong editing is to finetune a model on data as they arrive (Lee et al., 2020). However, finetuning on singular errors is prone to overfitting — even when using customized regularization techniques (Lin et al., 2022) — and the edited model catastrophically forgets its original training data, devaluing upstream pretraining. Further, edited models also forget previously-fixed errors, counteracting the objective of editing in the first place (Sinitsin et al., 2019). Another candidate for lifelong editing is through standard model editing which updates model weights with minimal influence on upstream data, outperforming finetuning methods Mitchell et al. (2022a). While a promising direction, these methods require large amounts of data in order to make edits. For example, some recent works use training sets filled with pre-collected errors to train hypernetworks that edit a model's behavior by predicting new weights (Mitchell et al., 2022a) or offloading predictions to a new classifier (Mitchell et al., 2022b). However, large pools of training edits are rarely available

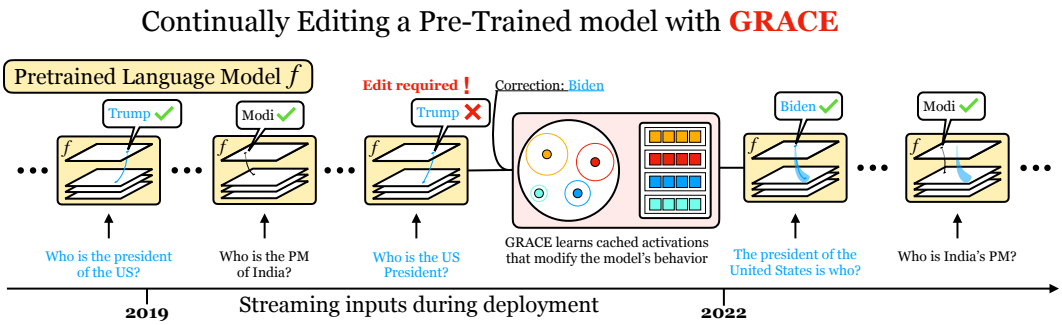


Figure 1: GRACE edits pretrained models during deployment by maintaining layer-specific codebooks that cache learned activations for selected layers. When new instances similar to previous edits arrive, the corresponding activations can be passed to the subsequent layer to make the model predict desired outputs. GRACE continuously learns to edit a model using *only* streaming errors identified during deployment without accessing any upstream data.

before deployment, otherwise the edits could be used during pretraining. Further, regularized finetuning approaches (Meng et al., 2022; Sinitsin et al., 2019; De Cao et al., 2021) rely on sets of semantically-equivalent inputs to preserve upstream model performance. Additionally, prior model editors (Mitchell et al., 2022a; Sinitsin et al., 2019; De Cao et al., 2021; Mitchell et al., 2022b) have yet to consider *sequential*. Consequently, they perform poorly in the sequential editing scenario, as shown in Section 3.

In this paper, we introduce *Lifelong Model Editing*. Given a model f_0 that was pretrained on upstream instances $\mathcal{U}$, let f_t denote the *edited* model at timestep t. When deploying f_0, we begin to observe a stream of errors made by the model $\{(X_t^e, y_t^e)\}_{t=1}^{T}$, where $f_{t-1}(X_t^e) \neq y_t^e \; \forall \, t$. At each step t, given an edit input and edit label pair (X_t^e, y_t^e) our aim is to produce an edited model f_t that meets three criteria:

1. $f_t(X_t^e) = y_t^e$; the instantaneous error is corrected.
2. $f_t(X_i^e) = y_i^e$ for $i \leq t$; the model remembers the corrections for previous errors.
3. $f_t(X) = y \; \forall \, (X, y) \in \mathcal{U}$; the model's upstream training performance is maintained.

Lifelong editing is particularly challenging when edit labels conflict with training labels and when the distributions of edits differs from the model's upstream training data.

To address these challenges of Lifelong Editing, we propose GRACE. GRACE modifies individual layers of a frozen, pretrained model f_0, treating it as an encoder. To modify f_0's predictions for a given input, GRACE uses the input to a selected layer as an encoding, with which it selects the nearest key from a codebook memory filled with previously-learned keys. Each key corresponds to an activation l that can be passed into the model's next layer, which decodes the values and ultimately makes a prediction. GRACE makes local edits to a large model's behavior by learning activations that influence the model's predictions for specific concepts maintained by separate keys. By managing the codebook size and learning small radii of influence per key, GRACE fixes model mistakes without decaying performance on upstream training data. For example, in Figure 1, GRACE corrects f's predictions about the United States President without impacting knowledge of India's Prime Minister. As new inputs and errors arrive over time, GRACE only needs to decide when to retrieve activations—determined by similarity search—and how to modify existing keys and values. GRACE thus provides a new paradigm for managing conflicts between upstream and downstream behavior. By monitoring a growing codebook, we can directly inject prior knowledge about how influential an edit should be. Further, if edits shift out of distribution, GRACE can naturally augment the complexity of a pretrained model by expanding its keys to fill new regions of an increased representation space.

Our contributions in this work are:

1. We cast model editing in a more realistic lifelong streaming setting. To our knowledge, this setting is unstudied, yet is crucial to successfully deploying large language models.

2. We present GRACE, a novel key-value model editor which learns to cache and retrieve activations for selected layers *using only errors observed during deployment*.
3. Our experiments show that GRACE is a state-of-the-art model editor, ultimately outperforming alternatives on real model editing tasks with shifting data and label distributions.

2 EDITING LARGE MODELS WITH GRACE

2.1 PROBLEM FORMULATION

Assume we are given a large model f that was pretrained on some upstream dataset $\mathcal{U}$. Let f_0 denote the frozen pretrained model at time step $t = 0$. We then deploy f_0 on some real task and begin monitoring its predictions $\hat{y}_t = f(X_t)$ as inputs X_t stream in, one step t at a time. Over time, we receive some errors X_t for which $\hat{y}_t \neq y_t$, the true label for X_t. In order to continue safely deploying f, we aim to *edit* f such that $f(X_t) = y_t$ – let f_t denote the *edited* model at step t. Note that f_t will be different from models used at prior steps and that prior models are discarded. Beyond correcting f_{t-1} on X_t, we also desire that f_t maintains high performance on 1) prior edits $X_{<t}$ and 2) the upstream dataset $\mathcal{U}$. Further, upstream training data are often proprietary or prohibitively large, so we assume no access to $\mathcal{U}$ during editing, in contrast to prior works' stronger assumptions.

2.2 GENERAL RETRIEVAL ADAPTORS FOR CONTINUAL EDITING

GRACE presents a novel paradigm for model editing: As errors are identified and corrected over time, GRACE modifies a pre-trained model's behavior *without* altering its weights. GRACE can edit any desired layer in a pre-trained model by learning a discrete codebook $\mathcal{C}$ that modifies a layer's behavior for similar instances. While inspired by recent success in learning discrete key-value models (Träuble et al., 2022), GRACE operates *within* the layers of a pretrained model, and introduces a method for managing keys and values over time **without** pretraining the codebook. A GRACE codebook added to layer l contains the following contents:

- *Keys* ($\mathbb{K}$): Set of keys which are the same size as f_0's activations from layer $l-1$.
- *Values* ($\mathbb{V}$): Set of values that are learned as the model is deployed and accumulating errors. Each value has the same number of dimensions as the next layer's input.
- *Influence radii* ($\mathcal{E}$): Each key is paired with an *influence radius* ϵ, which are used as a threshold for similarity matching. Given a GRACE layer l and f_0^{l-1}, the pretrained model activations at layer $l-1$, we use a similarity search over existing keys to find the key closest to f_0^{l-1} via a distance function $d(\cdot)$:

  $$d_{min} = \min_i (d(f_0^{l-1}, K_i^l)).$$

  GRACE is activated at layer l only if $d_{min} \leq \epsilon_k^l$, where k indexes the most similar key. The larger the value of ϵ, the more *influence* the key has, since it covers more of the embedding space. As we discuss below, as GRACE fixes edits over time, ϵ values will shrink and expand to ensure that GRACE layers adapt to changing data distributions.
- *Key labels* ($\mathbb{Y}$): When a new key is added, its corresponding edit label is also stored. By accessing edit labels *only* while editing, keys and their ϵs can be adapted to generalize to similar instances without influencing too much of the embedding space.

To perform inference with a GRACE-edited model, layer l is computed using a discrete key-value search over GRACE's keys:

$$f_t^l(\cdot) = \begin{cases} \text{GRACE}^l[f_0^{l-1}], & \text{if} \min_i (d(f_0^{l-1}, K_i^l) - \epsilon_i^l) > 0 \\ f_0^l, & \text{otherwise,} \end{cases}$$

where ϵ_i^l and K_i^l are the influence radius and key i in layer l, respectively. $d(\cdot)$ is a distance function (we use Euclidean distance in our experiments). By using a discrete similarity search, if a new input is unlike any cached keys, GRACE simply defers to f_0's pretrained weights. This way, if the edit instances are out-of-distribution for the training data, GRACE layers can avoid interference with

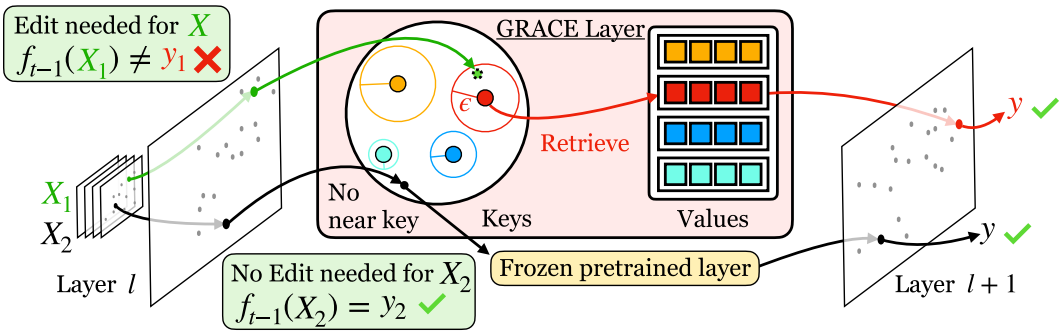


Figure 2: GRACE learns and maintains a codebook for selected layers in a pretrained model. Each codebook contains keys and their corresponding ϵ and values. When an edit like X_1 is identified, a similarity search is performed and either a new key is added or an existing key and its ϵ are updated. Unrelated instances like X_2, for which no edit is required, land far from existing keys and defer to the pretrained layer's predictions. If new instances are similar to previously-seen edits, GRACE can generalize to make predictions for them by retrieving their nearest key's activations.

upstream data by leaving the model weights unaltered. Further, edits are usually rare compared to streaming inputs, so a GRACE-edited model will often defer to a pretrained layer's outputs.

As illustrated in Figure 2, when an edit is required, f_0 serves as an encoder, computing an embedding for an instance at layer l. Then, f_0^l serves as a query across any existing keys in the GRACE codebook for layer l. A GRACE layer can perform one of the following operations at any given time step:

1. **KEY-INITIALIZE**: If no keys are present, GRACE initializes the query as a key. This step only happens when the model makes its very first error—otherwise, GRACE always either Adds a key or Splits an existing key.
2. **KEY-ADD**: If the f_0^{l-1} does not fall within the ϵ radius of any existing keys according to distance function $d(\cdot)$, then a new key is initialized to be f_0^{l-1} along with a corresponding value v, base influence radius ϵ^l, and edit label y^e.
3. **KEY-UPDATE**: If f_0^{l-1} is similar-enough to any existing keys—determined by whether or not the query lands inside the ϵ radius of any keys, we need to ensure that the matched key's activation carries the same semantic content as the edit input. To do this, we compare the current edit's label to the label cached along with the matched key:
   (a) *If the query's nearest key has the same label as the edit label*, **expand** the nearest key's ϵ to encompass the query:
   $$\{K_i^l : [V_i^l, \epsilon_i^l]\} \rightarrow \{K_i^l : [V_i^l, d(f_0^l, K_i^l)]\}$$
   (b) *If the query's nearest key has a different label from the edit label*, **split** the nearest key into two keys by (1) decreasing the influence radius of the nearest key, then (2) creating a new key-value pair where the key is the query. The new key is the activation of f_0 at layer $l-1$ for the current edit input.
   $$\{K_i^l : V_i^l, \epsilon_i^l\} \rightarrow \left\{ \begin{array}{rr} K_i^l : & [V_i^l,\ 0.5 * d(f_0^l, K_i^l)] \\ f_0^l\ : & [V_{i+1}^l,\ 0.5 * d(f_0^l, K_i^l)] \end{array} \right\}$$

As edits stream in, by continuously adding and updating GRACE's keys and values, the embedding space for a selected layer l becomes partitioned according to which instances need modified outputs. When *not* performing edits, these operations are bypassed, and keys are entirely frozen, regardless of whether or not the instance lands within a key's influence. Overall, GRACE introduces a new model editing paradigm in which edits can be made sequentially, similar edits are encouraged to be edited similarly, and the ultimate influence of new edits can be controlled and monitored explicitly.

2.3 Training and Inference with GRACE Layers

When an edit is required and GRACE layers are used, the closest key matching the edit input query is found using a similarity search followed by the key update steps described above. Once a closest

key is either found or newly created, the corresponding value is returned. As illustrated in Figure 2, this value serves as the activation of the next layer. Then, the final model output is computed and a downstream loss is calculated between the predicted label and the edit label. During backpropagation, only the values of the GRACE layers are updated based on the downstream loss, thereby learning activations that alter the model's predictions.

During inference, at each GRACE layer, the key closest to the query embedding is identified through a similarity search. In our experiments, we find that euclidean distance works well, similar to (Träuble et al., 2022). The corresponding value is returned as an activation to the next layer, and this step is repeated for all GRACE layers added to the model. GRACE naturally applies to all recent transformer models, as any layers that compute new updates can be edited using it.

3 Experimental Setup

To evaluate GRACE, we investigate three points. First, using synthetic data, we validate that GRACE indeed learns local fixes for pre-trained models. Second, using a real Question Answering task, we evaluate GRACE's capacity to alter a language model's predicted tokens over long deployment periods. Third, we experiment with a real label-shift case, evaluating lifelong model editing when label distributions change during deployment. We compare GRACE with three alternatives:

Finetuning. First, we finetune a model's weights on errors as they arrive, akin to an error-only version of previous online finetuning (Roberts et al., 2020). As edits can be rare during deployment, finetuning methods easily overfit.

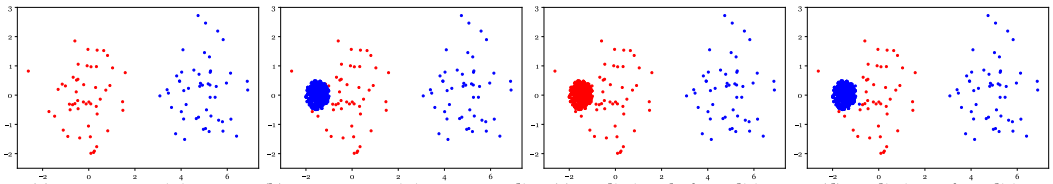
MEND. As a representative of recent hypernetwork approaches, we also implement a streaming version of MEND (Mitchell et al., 2022a). As proposed, MEND pretrains a model edited on a large set of edits before being deployed to make an edit to a large model. Further, they regularize this pretraining with sets of semantically-equivalent instances for each edit and samples from the upstream training dataset. However, in our setting, we assume no access to upstream data, upstream edits, or similar instances. Therefore, we train a MEND network alongside a deployed pretrained model as edits stream in. When a new instance arrives, MEND's hypernetwork is finetuned slightly to predict accurate low-rank updates to the pretrained model's weights. Then, the MEND-predicted updates are applied. This way, over many edits, knowledge of previous edits are baked into the MEND model. However, by updating all weights each time based on no training data, this method is not expected to perform well, as we show experimentally, referring to this adapted version as Streaming MEND.

Memory Network. Finally, we compare against a memory network-based adaptor, picking a number of embeddings and using the pretrained model's embeddings as input to a small memory module. The memory module predicts one attention value per embedding, then takes a weighted sum of prior embeddings which then serves as input to the next layer. This comparison serves as a softer version of our discrete method. With access to only streaming errors, this approach is also prone to overfit.

Implementation Details We experiment with T5 (Raffel et al., 2020) and BERT (Devlin et al., 2018), though GRACE is general and applicable to other large NLP models as well. To ensure fair comparisons, we edit the same layers across all methods for each experiment. For T5 experiments, we edit the dense-relu-dense layer of the seventh encoder block of a 60 million parameter model.

During evaluation, we hyperparameter tune each method, only considering methods that successfully edit each input instance. For Finetuning, Memory Networks, and MEND, we edit with learning rates of $1e^{-1}$, $1e^{-2}$, $1e^{-3}$, $1e^{-4}$, and $1e^{-5}$. Intuitively, small learning rates retain upstream performance while large learning rates learn to apply new edits. Therefore, for some learning rates, finetuning methods may achieve higher individual upstream and online scores than we report. However, akin to constrained optimization, the minimal editing success is to actually modify the model's performance on one edit. Here, finetuning is particularly prone to overfitting.

Metrics To evaluate lifelong model editors, we track two key metrics over time: (1) **Upstream performance** is the model's performance on a portion of its training set, indicating how much the model remembers its past knowledge, and (2) **Online performance** is the edited model's accuracy on the history of streaming edits. Both should ideally be high, though they often contradict in practice.



(a) Upstream Training Data (b) Upstream Training Data + Edits (c) Predictions before editing (d) Predictions after editing

Figure 3: Results for GRACE on synthetic data. A model is pretrained on the data in (a), then label shift is introduced in (b) via a local set of label-swapped edit instances. Then, the pretrained model predicts the wrong class for these edit instances, as shown in (c). Finally, as shown in (d), GRACE successfully edits the pretrained model, swapping its labels without impacting distant data.

4 Experiments

4.1 Illustrating GRACE with Synthetic Data

Experiment: We first show GRACE's performance in a simple and illustrative synthetic dataset. As shown in Figure 3(a), we sample two-dimensional instances from two clusters, each corresponding to a class and shown in red and blue. We then use this dataset to pretrain an upstream three-layer fully connected classification network. The network projects the instances into 100 dimensions, 100 dimensions, and back to 1 dimension for binary classification, using Rectified Linear Units (Nair & Hinton, 2010) as intermediate activation functions. Since these data are linearly separable, the model then learns to split the feature space vertically. We then introduce a set of likely edits by introducing a tight circle of instances inside the distribution on the left, simulating local label shift, as shown in Figure 3(b). As shown in Figure 3(c), the upstream model then classifies the flipped labels incorrectly. We then use a GRACE layer to edit the pretrained model, sampling edits from the edit set one by one and initializing ϵ to be 0.45 to roughly align with the scale of the edit set.

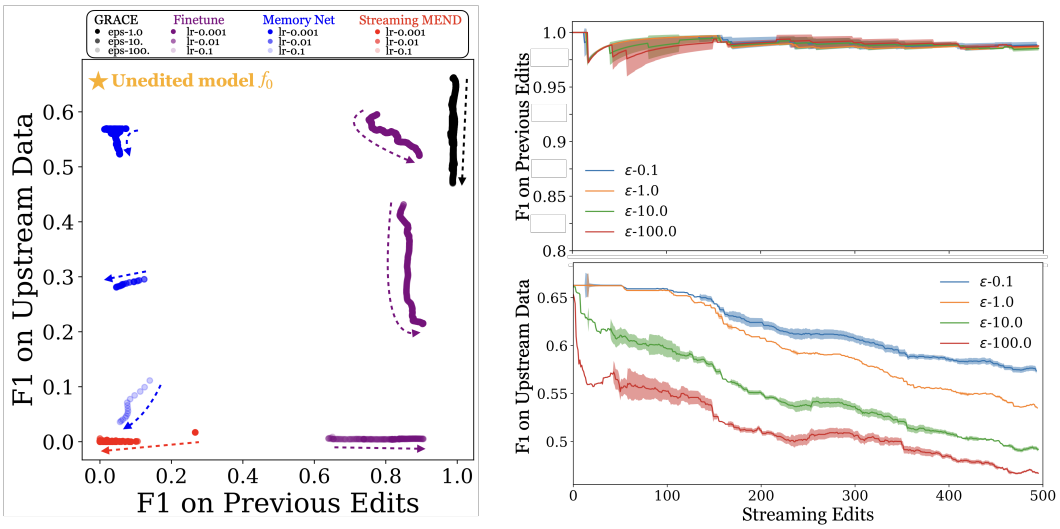
Result: In Figure 3(d), we see that GRACE succeeds to locally edit the model's predictions for the edits without influencing the model's predictions on unrelated training data. While this case is relatively simple, Finetuning a model only on errors—especially those from a single class—leads to catastrophic forgetting on upstream data. We show in the Appendix that Finetuning indeed flips all predictions to the blue edit class. Additionally, GRACE edits this model using only one key. Using one key is ideal in this simple setting, since the edits are clustered. As we show next, as tasks become more complicated, edits spread out and require more keys to cover the model's representation space.

4.2 Question Answering with Shifting Answers

Experiment: Pretraining and finetuning large language models is an increasingly-successful approach to Question Answering (QA) (Roberts et al., 2020), especially for open-domain tasks (Prager et al., 2007). In recent methods for open domain QA, language models learn to correctly answer questions without any context (Izacard & Grave, 2021). While such approaches are growing more successful, as the world changes the correct answers to open-domain questions naturally change. For example, when asking "Who is the president of the United States?" the correct answer will change every few years. As correct answers shift, pretrained models age and underperform (Meng et al., 2022).

We study methods for correcting for QA shift through the lens of model editing. We edit a T5 model (Roberts et al., 2020) that was pretrained on the Natural Questions dataset (NQ) (Kwiatkowski et al., 2019) to correct its mistakes on questions from the downstream zsRE dataset (Levy et al., 2017) using splits from De Cao et al. (2021). For repeated upstream evaluation, we sample 1000 instances from NQ. For editing, we sample 1000 edits from zsRE, including random samples of 5 rephrasings for each question. We focus our QA evaluation on the standard F1 metric, measuring overlap between predicted and true tokens.

Before any editing, T5 achieves an F1 score of 0.66. As zsRE questions stream in, we first check that the model makes a mistake. We run this check for every instance in this experiment as the model is being updated frequently over time. If T5 makes a mistake, we then pass it to GRACE to edit the model. Since edits stream in, errors are identified with respect to only the current version of the model



(a) Comparing all methods' performance on upstream data and previous edits. Ideal performance is in the upper right corner. Arrows denote each model's performance throughout streaming. GRACE consistently outperforms the alternative approaches to lifelong editing, achieving high F1 on previous edits while preserving upstream F1.

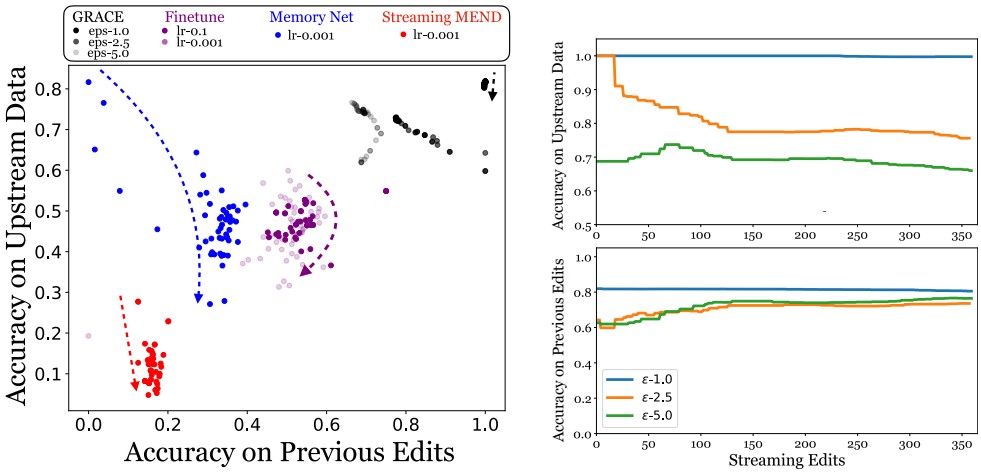
(b) Impact of parameter ϵ on GRACE. By increasing ϵ, more upstream data lands within the radius of influence for any given key, decaying upstream F1. As expected, each ϵ still achieves perfect F1 on all previous edits, indicating that edits are distant from one another in this dataset. Standard deviations are computed over three repetitions.

Figure 4: Results for lifelong editing for context-free QA editing. Each editor sequentially updates the same pretrained model on around 500 errors sampled from the zsRE dataset. GRACE achieves strong F1 scores on both upstream data and previous edits, and ϵ exerts direct control over this trade-off.

at timestep t. Over 1000 edits, each method ends up making errors on about 500 inputs. After each edit, we report the new model's performance on a holdout set of 1000 questions from the upstream Natural Questions dataset. An ideal model editor should only update the requested edits, leaving unrelated instances untouched. We also report Online F1: the edited model's F1 score on the growing set of previous edits. A successful model editor should make new edits without forgetting old edits.

Result: As we show in Figure 4a, GRACE successfully edits the pretrained T5 model while remembering upstream information. For the Finetuning, Memory Network, and MEND baselines, we vary the learning rate. For GRACE, we vary ϵ, the initial radius of influence for keys as they are initialized. As each method is trained on streams of edits, we show arrows indicating their progression through the multi-objective space over time. The optimal performance is in the upper right corner. Indicated in black, GRACE succeeds to maintain high F1 scores on upstream data, while also remembering previous edits. As described above, this is by adding a small number of new parameters in the GRACE codebook. However, as discussed in Section 4.3, the number of new parameters is tiny compared to the pretrained model. As expected, as learning rates increase for Finetuning and the Memory Network, their F1 performance on upstream data severely decreases. As finetuning progresses, the models move farther from their initial state. Further, indicated by the arrows pointing to the lower right, the Finetuning succeeds to generally increase F1 on all previous edits over time, indicating that.

As reported in Figure 4b, we also show GRACE's editing performance throughout editing for different values of ϵ. ϵ controls the influence radius of each stored key in a layer's GRACE codebook. Larger ϵ values result in few keys that cover large portions of the embedding space. As expected, this leads to decayed upstream performance, since more upstream data will conflict with edits. However, this may be a boon: GRACE provides an intuitive interface for replacing the knowledge of an upstream model, which may have impacts on de-biasing pretrained models. We also notice that each ϵ leads to near-perfect F1 on previous edits, indicating a perfect memory. This may occur when edits are far apart in the embedding space and we hypothesize that F1 on previous edits will eventually drop as the number of edits increases. Still, even performing 500 edits in a row is a substantial improvement over the state-of-the-art alternatives.



(a) Method comparison on Annotation Relabeling using the SCOTUS dataset. Similar to Figure 4, the upper right corner is ideal and we find that GRACE outperforms alternative methods on both axes.

(b) Impact of parameter ϵ on GRACE on the SCOTUS dataset. As ϵ increases, we can observe that GRACE provides a natural tradeoff between upstream accuracy and the accuracy on the streaming edits.

Figure 5: Results for lifelong editing for the annotation relabeling task. Each editor sequentially updates the same pretrained model on around 400 errors sampled from the Fairlex Scotus dataset.

4.3 Correcting for Shifting Annotation Guidelines

Experiment: Labels are often collected via annotation guidelines. However, over time, these guidelines may shift, so label accuracy changes. We experiment with correcting for shifting annotations using the Fairlex benchmark (Chalkidis et al., 2022), specifically the SCOTUS subset. SCOTUS contains documents outlining the US supreme court's opinions on court cases. Given an opinion, the goal is to predict a case's "issue area". Each document is labeled with one of 11 issue areas and SCOTUS is divided into a training split with 7.4k cases from 1946-1982 and a test split with 931 cases from 1991-2016. While SCOTUS labels do shift, we exacerbate the change, inducing label shifts by merging semantically-similar labels in the training split. For example, we turn {Civil Rights, First Amendment} into one class {Civil Rights}. In the test split, we then separate the labels.

We then study each method's capacity to correct mislabeled documents. Pretrained models without any editing or finetuning are expected to severely underperform in this strict setting because training data are unrepresentative. We finetune a BERT sequence classification model (Devlin et al., 2018) on 90% of the training data for 10 epochs with a learning rate of $1e^{-5}$, holding out 10% for validation, and use the test split as our edit set. The accuracy of this finetuned model is 82% on the held out validation set while it is 55% on the edit set, before any model editing. This huge drop in performance further validates the challenging setting. Over 931 potential edits, each model editor performs roughly 400 edits, since edits are only made when the model makes a mistake.

Result: As shown in Figure 5, GRACE achieves a better trade-off between successfully editing BERT and remembering upstream information, compared to other baselines. For the Finetuning, Memory Network, and MEND baselines, we vary the learning rate while for GRACE, we vary ϵ, the initial influence radius for keys as they are initialized. Similar to the results in Section 4.2, we illustrate the performance of different approaches over edit streams in Figure 5a using arrows indicating their progression through the multi-objective space over time. Indicated in black, GRACE maintains high accuracy on upstream data, while remembering previous edits. As expected, as learning rates change, the other methods' performance drops severely. As finetuning progresses, the alternatives stagnate for both objectives. As indicated by the upward trending arrow, finetuning with lower learning rate begins to show good performance on upstream data while sacrificing online accuracy.

Figure 5b shows the effect of ϵ on both metrics as the model is deployed on the edit stream. With a small $\epsilon 1$ of 1, GRACE achieves nearly-perfect online accuracy and high upstream accuracy, indicating that GRACE codebooks memorized the edit stream. As many edits arrive, such behavior could result in massive codebooks. As we increase ϵ, we begin to see that GRACE trades off upstream accuracy for online accuracy. Thus, GRACE layers provide a natural way for balancing upstream accuracy with

ϵ	Num edits	Num keys	key / error ratio	Upstream Accuracy	Online Accuracy
1.0	403	403	1.0	79.8 %	99.7%
2.5	387	256	0.66	73.9%	74.2%
5.0	375	245	0.65	76.5%	64.6%

Table 1: Effect of ϵ on the number of keys created by GRACE for Shifting Annotation Guidelines.

the online accuracy by performing spot fixes in the model's activation space while preserving model behavior on related semantic concepts.

We further demonstrate the effect of ϵ on the number of keys and resultant performance metrics, as shown in Table 1. Here, we see that as ϵ increases, the number of keys decreases and the codebook becomes more compact. An optimal ϵ will balance upstream and online accuracy.

5 Related Work

Model editing. Model editing is a recent and active research area. Classic approaches focus on focus on regularized-finetuning, altering a model's weights by incorporating auxiliary information, like upstream training data and semantically-equivalent instances (Sinitsin et al., 2019). Recent works have extended this paradigm to pretrain hypernetworks that predict edits (De Cao et al., 2021; Mitchell et al., 2022b;a), often decomposing weight updates into low-rank components (Meng et al., 2022; Hu et al., 2022). Due to notorious training costs, many works focus on editing transformer models (Zhu et al., 2020), leading to natural extensions to multi-lingual editing (Xu et al., 2022). Many recent works have considered parameter-efficient finetuning, often by using low-rank updates Hu et al. (2022). While parameter-efficient approaches such as prompt tuning train an extension of pretrained mdoels, they often require more training steps and are prone to overfitting than regular finetuning (Zhong et al., 2022; Su et al., 2022), making them inappropriate from a model editing standpoint compared to approaches such as Mitchell et al. (2022a) and Mitchell et al. (2022b).

Recent works have also begun considering continual finetuning, where large language models are refined over time as new instances arrive. For example, Lin et al. (2022) show that regularizing finetuning with continual learning methods like Elastic Weight Consolidation (Kirkpatrick et al., 2017), Experience Replay (Rolnick et al., 2019), and Maximally Interfered Replay (Aljundi et al., 2019) decay upstream performance rapidly when shifting between multiple tasks.

Key-Value Methods for Continual Learning. Key-value methods have arisen as a powerful paradigm for a variety of machine learning problems. These approaches have deep roots in computer vision (Van Den Oord et al., 2017; Liu et al., 2021), driving recent high-profile results like DALLE-2 (Ramesh et al., 2022). Further, key-value methods are particularly strong on continual learning settings, with recent works demonstrating prompt-learning for NLP (Wang et al., 2022b;a) and applications like text retrieval (Xiong et al., 2021). Recent works have shown that *discrete* key-value methods in particular performs well on distribution shifts (Träuble et al., 2022), with recent works extending to question answering (Dai et al., 2022). This is because the stored values can remain within the expected distribution of a downstream encoder, regardless of shifting inputs. We corroborate this finding in our experiments, where we demonstrate GRACE's robustness to shifting inputs and labels.

6 Conclusions

Language models are quickly becoming larger and are being applied to a diverse set of downstream tasks. However, they are often computationally prohibitive to finetune and easily forget past knowledge. In this work, we proposed a realistic problem setting where we edit such large models, *Lifelong Model Editing*, and presented GRACE, a plug-in module that can wrap around any given layer in a large pretrained model. GRACE layers (1) retain the functionality of the original model, thereby minimizing catastrophic forgetting and (2) adapt to changing data distributions by storing a codebook of cached activations that can grow or shrink over time. We demonstrate GRACE's efficacy by showing that GRACE provides the best trade-off between upstream performance and accuracy on streaming edits among competing model editing baselines and finetuning methods.

7 REPRODUCIBILITY

To encourage reproducibility of both GRACE and our experiments, we have provided thorough descriptions of our experimental setup in Section 3. Our descriptions contain specific layers we choose to edit. In Section 2, we include clear steps that detail GRACE's behavior.

8 ETHICAL CONCERNS

Model editing may alleviate some ethical concerns, for example correcting previously-incorrect information. However, editing models may also introduce new errors if the proposed fixes are themselves incorrect or biased against a given sub-population. Researchers who design, implement, and deploy such techniques should consider how to best evaluate sources of edits and verify they are not harmful.

REFERENCES

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. *ArXiv preprint, abs/1908.04742*, 2019.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Ilias Chalkidis, Tommaso Pasini, Sheng Zhang, Letizia Tomada, Sebastian Felix Schwemer, and Anders Søgaard. Fairlex: A multilingual benchmark for evaluating fairness in legal text processing. *arXiv preprint arXiv:2203.07228*, 2022.

Yi Dai, Hao Lang, Yinhe Zheng, Fei Huang, Luo Si, and Yongbin Li. Lifelong learning for question answering with hierarchical prompts. *arXiv preprint arXiv:2208.14602*, 2022.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *EACL 2021-16th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 874–880. Association for Computational Linguistics, 2021.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov.

Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*, 2020.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, 2017.

Bill Yuchen Lin, Sida I Wang, Xi Lin, Robin Jia, Lin Xiao, Xiang Ren, and Scott Yih. On continual model refinement in out-of-distribution data streams. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3128–3139, 2022.

Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *arXiv preprint arXiv:2202.05262*, 2022.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022a.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pp. 15817–15831. PMLR, 2022b.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

John Prager et al. Open-domain question–answering. *Foundations and Trends® in Information Retrieval*, 1(2):91–231, 2007.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. Editable neural networks. In *International Conference on Learning Representations*, 2019.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, et al. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3949–3969, 2022.

Frederik Träuble, Anirudh Goyal, Nasim Rahaman, Michael Mozer, Kenji Kawaguchi, Yoshua Bengio, and Bernhard Schölkopf. Discrete key-value bottleneck. *arXiv preprint arXiv:2207.11240*, 2022.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. *arXiv preprint arXiv:2204.04799*, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022b.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021.

Yang Xu, Yutai Hou, and Wanxiang Che. Language anisotropic cross-lingual model editing. *arXiv preprint arXiv:2205.12677*, 2022.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. Panda: Prompt transfer meets knowledge distillation for efficient model adaptation. *arXiv preprint arXiv:2208.10160*, 2022.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

A LIMITATIONS

While lifelong model editing is new and challenging, our method GRACE has limitations. Understandably, introducing codebooks between layers of a pretrained model slows down inference. This slow-down is largely due to repeated similarity searching between embeddings and keys. Even though GRACE's current design does not emphasize inference time, making GRACE faster is a promising avenue for future work, especially given recent accelerations in adaptor methods like LoRA (Hu et al., 2022) and language models using similarity like Gopher (Rae et al., 2021).

While GRACE can scale to use in multiple layers simultaneously, we do not perform these experiments. In the most-general case, GRACE layers may be stacked between subsequent layers. Then, the output of one codebook may be passed to another codebook. Stacking codebooks will lead to highly-flexible models that provide friendly interfaces with which to control their outputs. Our work takes the first step towards this new model primitive, which has clear implications for responsible machine learning.

Finally, autoregressively decoding long sequences with GRACE is particularly challenging. In the long term, language model editors will control long sequence generation. While this is a current challenge for all current editors, GRACE will require modification to generate many tokens in a row. This limitation is due to GRACE's similarity-search between input queries, which are embeddings, and existing keys: During autoregressive generation, the embedding for prior tokens is updated. Then, the resultant embedding may point to new GRACE keys, leading to continuous distribution shift.