Measuring and Manipulating Knowledge Representations in Language Models

Evan Hernandez [1] **Belinda Z. Li** [1] **Jacob Andreas** [1]

Abstract

Neural language models (LMs) represent facts about the world described by text. Sometimes these facts derive from training data (in most LMs, a representation of the word *banana* encodes the fact that bananas are fruits). Sometimes facts derive from input text itself (a representation of the sentence *I poured out the bottle* encodes the fact that the bottle became empty). Tools for inspecting and modifying LM fact representations would be useful almost everywhere LMs are used: making it possible to update them when the world changes, to localize and remove sources of bias, and to identify errors in generated text. We describe REMEDI, an approach for querying and modifying factual knowledge in LMs. REMEDI learns a map from textual queries to fact encodings in an LM's internal representation system. These encodings can be used as *knowledge editors*: by adding them to LM hidden representations, we can modify downstream generation to be consistent with new facts. REMEDI encodings can also be used as *model probes*: by comparing them to LM representations, we can ascertain what properties LMs attribute to mentioned entities, and predict when they will generate outputs that conflict with background knowledge or input text. REMEDI thus links work on probing, prompting, and model editing, and offers steps toward general tools for fine-grained inspection and control of knowledge in LMs.

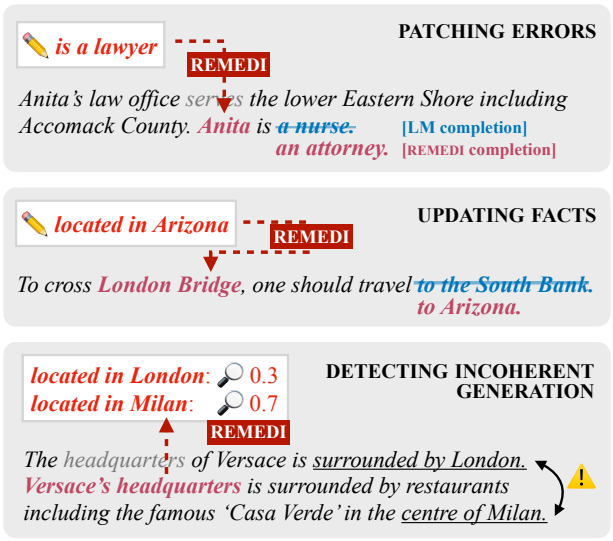

Figure 1. REMEDI can patch errors made by an LM and insert new facts with or without context provided in the prompt. It can also help detect errors before generation.

1. Introduction

Neural language models (LMs) build implicit, structured models of the state of the world: their representations encode general knowledge (Petroni et al., 2019) and situations described in input text (Li et al., 2021). Sometimes these representations contain mistakes: they can be inaccurate or incoherent, resulting in errors in generated text (Fig. 1). Even as LMs improve, versions of these problems are likely to persist: large LM training sets contain erroneous and contradictory information, go out of date, and harbor unexpected biases (Bender et al., 2021). And even in domains where LM generation is more reliable, understanding how model-internal representations relate to generation is crucial for attribution and controlled generation (Akyürek et al., 2022; Dai et al., 2022). There is thus a fundamental need for techniques that can *measure* and *manipulate* LMs' knowledge about the world in general and the world as described in specific documents.

This paper introduces REMEDI (REpresentation MEDIation), a technique for doing both. REMEDI discovers directions in representation space corresponding to encodings of factual attributes (like *is a lawyer* in Fig. 1). When these encodings are added to LMs' representations of entities (like *Anita*), they *edit* the facts that models attribute to those entities—in some cases causing LMs to generate output that cannot be produced with a corresponding textual prompt. Encodings produced by REMEDI can also be used to *interpret* LM representations, making it possible to probe LMs' factual knowledge, and to predict when they will generate incorrect or incoherent output.

[1]MIT CSAIL. Correspondence to: EH <dez@mit.edu>.

The effectiveness of REMEDI across applications involving context-specific and general background knowledge shows that neural LMs represent and integrate information from these two knowledge sources in a unified manner. REMEDI offers steps towards tools that can monitor and control language generation by interacting with these representations directly, specifying facts and situations in an LM's native encoding scheme. Our code and data are publicly available.[1]

2. REMEDI

Motivations: control and interpretability Consider the examples from Fig. 1 (top). In the first example, the LM is **prompted** with the text *Anita's law office serves the lower Eastern Shore...*, which provides some **context** about the **entity** Anita. However, when the LM generates a continuation of this prompt, it asserts that Anita is a nurse, an assertion that is incoherent with the preceding context. We term this incoherence a failure of **context mediation**: information provided in the textual context has failed to mediate the LM's predictions. It would be useful to be able to identify and fix such errors, changing a model's encoding of entities like *Anita* to ensure that she is correctly described as an *attorney*. In addition to ensuring discourse coherence, it is often desirable to modify prior associations in LMs. In Fig. 1 (middle) the LM strongly associates *London Bridge* with the city of *London* because the most famous London Bridge is located there. However, there could be (and are[2]) other London Bridges, and we might wish to control an LM to make the lesser-known bridge more salient.

It is sometimes possible to achieve these goals by carefully prompting models with the right input text. But due to the non-systematic opaque nature of prompt engineering (Jiang et al., 2020b), significant manual effort is often required to find a prompt (if one exists at all) that yields correct behavior and generalizes to different use cases. Fig. 1 (bottom) highlights one instance of this challenge: though the LM is prompted with information that Versace headquarters is located in London, it still generates text consistent with a headquarters in Milan.[3] Techniques for localizing these failures within LMs' internal representations would make it possible to detect them in advance, and guide research aimed at mechanistic understanding of the relationship between LMs' internal representations and their textual outputs.

Overview At a high level, our proposed approach learns how to intervene in an LM's representation space to modify the LM's knowledge about a mentioned entity (like *Anita* in Fig. 2). This intervention ultimately updates the LM's representation of the entity to encode an **attribute** (e.g., *is a lawyer*) so that the LM will generate text about the entity consistent with the new attribute. This update operation can be specified by a single vector, and is applied to the hidden representation of a single token at a single layer. Representation edits produced by REMEDI can not only address failures of context mediation, but also build entities with desired properties from scratch (enabling controlled generation without textual prompts). Additionally, by comparing edit vectors to unedited representations, REMEDI makes it possible to inspect representations of entities and attributes produced during ordinary model operation.

Editing representations Assume we have a language model $p_{\text{LM}}(x)$ that assigns probabilities to strings x consisting of tokens $x_{1:n}$. In this paper, p_{LM} will always be an autoregressive transformer (Vaswani et al., 2017) pretrained on English text, as in the GPT family of models (Radford et al., 2019; Brown et al., 2020). These models decompose $p(x)$ into a product of next-token distributions given left context: $p_{\text{LM}}(x) = \prod_i p_{\text{LM}}(x_i \mid x_{1:i-1})$. Our goal is to insert a hidden state into the model that causes it to generate desired text about a target entity.

Where and how should we insert this new hidden state? Prior work suggests that LMs encode factual information in hidden representations of entity mentions. Attributes such as entity state (Li et al., 2021), perceptual properties (Abdou et al., 2021), and other semantic associations (Grand et al., 2018) have been shown to be *linearly decodable* from entity representations. This suggests that, to ensure that an LM encodes the fact *Anita is a lawyer*, it should suffice to find an appropriate transformation of the representation of the token *Anita*.

Formally, we denote the transformer's hidden representation for token x_i in layer ℓ as $h_i^{(\ell)}$, and we write $p_{\text{LM}}(x \mid h_i^{(\ell)} = z)$ to mean the output of p_{LM} with $h_i^{(\ell)}$ "hard-coded" to equal z during the forward pass.[4] Given representations of the entity h_{entity} and the target attribute h_{attr}, REMEDI specifically learns an **affine transformation** F that returns a new entity representation z according to:

$$z = F(h_{\text{entity}}, h_{\text{attr}}) = h_{\text{entity}} + W h_{\text{attr}} + b\,. \qquad (1)$$

such that when z replaces the entity inside the LM, the LM will generate text consistent with the target attribute.

How should we pick h_{attr}, W and b? Building on the success of linear probing approaches (Conneau et al., 2018; Belinkov & Glass, 2019), it is tempting to begin by training

[1] https://github.com/evandez/REMEDI

[2] Such as the one in Lake Havasu City, Arizona.

[3] These issues are not solved with scale: "prompt injection attacks" that cause LMs to ignore initial instructions (Perez & Ribeiro, 2022; Greshake et al., 2023) may also be viewed as failures of context mediation, and might (beyond the scope of this paper) also be mitigated with better tools for directly manipulating representations of tasks rather than facts.

[4] Henceforth we will omit the layer index, but the reader should always assume all operations occur at a single layer.

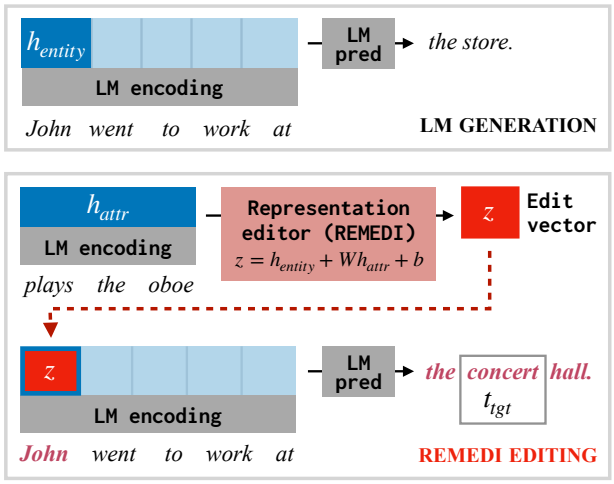

Figure 2. Illustration of REMEDI. Given an prompt (*John went to work at*) and a desired attribute (*plays the oboe*), REMEDI constructs an edit to the hidden representation of *John* that increases the probability of an appropriate completion (*the concert hall*).

a *classifier* for the presence or absence of attributes. For example, following Li et al. (2021), we could take h_{attr} to be the LM's own representation of the text of an attribute (like *plays the oboe*; Fig. 2), then optimize W and b to predict whether an entity representation encodes the attribute:

$$p(\text{attribute} \mid \text{entity}) = \sigma(h_{\text{entity}}^{\top} W h_{\text{attr}} + b)\ . \quad (2)$$

However, even when an LM encodes information in its representations, this information may not *causally influence* subsequent generation (Ravfogel et al., 2020; Elazar et al., 2021; Ravichander et al., 2021). An effective editor must specifically identify edits that are causally linked to the LM's generations.

Learning effective edits Instead of optimizing W and b to act as a classifier, REMEDI optimizes directly for their effect as an intervention to the LM. We assume access to a dataset of tuples $(x_{1:n-1}, i_{\text{entity}}, t_{\text{attr}}, t_{\text{tgt}})$, where $x_{1:n-1}$ is a textual **context** (e.g. *John went to work at*), i_{entity} is the index of an entity within the context, t_{attr} is the text of the attribute to be inserted (*plays the oboe*), and t_{tgt} is a **generation target**: a completion that should be assigned high probability if the attribute is correctly applied to x_{entity} (*the concert hall*). Following Li et al. (2021), we obtain a vector representation h_{attr} by averaging the LM's encoding of t_{attr}. We then train the editor F to maximize the probability that p_{LM} assigns to t_{tgt} after modifying the hidden representation of x_{entity}:

$$\mathcal{L}_{\text{tgt}}(z) = -p_{\text{LM}}(x_n = t_{\text{tgt}} \mid x_{1:n-1}, h_{\text{entity}} = z)\ . \quad (3)$$

See Fig. 2 for a visualization.

Learning Non-Destructive Edits When LMs encode strong prior associations between entities and properties (e.g., in the *London Bridge* example; see Hase et al., 2021), it is necessary to remove these facts while inserting new ones. To do so, we obtain a target string t_{prior} that the LM assigns a high pre-edit probability to, and optimize a loss term that encourages F minimize the probability of that token:

$$\mathcal{L}_{\text{prior}}(z) = p_{\text{LM}}(x_n = t_{\text{prior}} \mid x_{1:n-1}, h_{\text{entity}} = z)\ . \quad (4)$$

Finally, to prevent the degenerate solution in which the language model always (and only) predicts t_{tgt}, we penalize the language model for changing its distributions on all tokens between the entity mention and the time at which it predicts t_{tgt}:

$$\mathcal{L}_{\text{KL}}(z) = \sum_{x_i}^{x_i \neq x_{\text{entity}}} D_{\text{KL}}\Big(p_{\text{LM}}(\cdot \mid x_{<i},\ h_{\text{entity}} = z) \,\Big\|\, p_{\text{LM}}(\cdot \mid x_{<i})\Big). \quad (5)$$

Unlike the distribution over tokens at the end of the prompt, which should change dramatically under the intervention, the distribution over these intermediate tokens should not change significantly. $\mathcal{L}_{\text{KL}}$ penalizes such changes.

The complete objective function that REMEDI optimizes is:

$$\mathcal{L}(z) = \mathcal{L}_{\text{tgt}}(z) + \lambda_1 \mathcal{L}_{\text{prior}}(z) + \lambda_2 \mathcal{L}_{\text{KL}}(z)\ , \quad (6)$$

where λ_1 and λ_2 are hyper-parameters.

We evaluate REMEDI by studying its ability to control model output (Section 4) and to interpret and predict model behavior (Section 5).

3. Related Work

Probing factual knowledge Large language models (LLMs) trained on massive text datasets have been shown to encode context-agnostic factual knowledge, which can be queried through a text prompt (Petroni et al., 2019). Most work on extracting background factual knowledge from LMs focuses on designing textual *queries* for different sources of knowledge (Richardson & Sabharwal, 2020; Peng et al., 2022). Indeed, probes may sometimes recover factual information even in cases when LMs do not generate truthful outputs with high probability (Burns et al., 2022).

Probing representations of individual situations Neural LMs have also been shown to build representations of context-dependent knowledge. Li et al. (2021) show that they track aspects of entity state over a discourse, and this state can be extracted from LM representations of contextualized entity tokens. Furthermore, many LMs have been (indirectly) evaluated on their ability to track context-dependent knowledge by having their performance measured on downstream *reading comprehension* tasks in wich the LM is expected to answer questions about facts within a discourse. Reading comprehension datasets such

as CoQA (Reddy et al., 2019), RACE (Lai et al., 2017), and SQuAD (Rajpurkar et al., 2016) are now part of the standard evaluation suite for new LMs; and most modern LMs perform well (Brown et al., 2020). However, generating does not always imply *knowing*. Datasets contain spurious correlations (Gururangan et al., 2018), and LMs are sensitive to the phrasing of prompts and questions (Jiang et al., 2020b).

Editing LLMs In the past, LLMs have been predominantly adapted to new tasks and knowledge through fine-tuning (Devlin et al., 2019). Recently, with very large LMs, new classes of adaptation methods have been introduced, which generally fall into one of the following two categories: (1) *Prompt design* approaches prepend a textual prompt to each example specifying the adaptation target (Brown et al., 2020). Though these techniques have generally been explored in the context of teaching LMs *new tasks*, they have also been used to imbue LMs with new knowledge. (2) *Prefix-tuning* approaches prepend continuous learned tokens ahead of each example. The learned tokens specify a task for the LM similar to how a textual prompt would (Li & Liang, 2021; Lester et al., 2021). *Control token* approaches similarly use these learned tokens to controls aspects of LM generations, including sentiment (Dathathri et al., 2020), style (Keskar et al., 2019), and fine-grained semantics (Ross et al., 2022). Prompts can be fragile; LMs may fail to generate text consistent with the prompt, as shown in Fig. 1.

Finally, a large body of work examines how to localize and edit factual information in the LM's weight space (Meng et al., 2022a;b; Mitchell et al., 2022; Dai et al., 2022). For example, ROME (Meng et al., 2022a) introduces a technique to localize factual knowledge in LMs to a particular subset of MLP modules, and edits specific facts in a targeted way through rank-one modification of MLP weights. Unlike REMEDI, these approaches operate on models' weight matrices rather than representations, meaning they correct errors in models' background knowledge but not information provided in context.

4. Controlling Generation

We begin by showing that the REMEDI procedure described in Section 2 is an effective tool for *controlling LM output*. Intuitively, if REMEDI succeeds in creating a new entity representation encoding the desired attribute, text generated by the LM about the entity should at minimum (a) prefer generations consistent with the target attribute over potentially contradictory attributes and (b) remain as fluent as the original generations. Our experiments in this section test properties (a) and (b), as well as other quality measures, in two different settings. In the first setting, we use REMEDI to patch incoherence errors like the *Anita* example in Fig. 1, editing the LM to reinforce the information provided in the context. In the second setting, we use REMEDI to update factual associations about famous entities (such as the *Versace Headquarters* example in Fig. 1). These experiments show that REMEDI often successfully controls model behavior even when ordinary textual prompting fails. It can thus serve as a building block for future controlled generation interfaces that allow users to directly steer model behavior in representation space.

4.1. Patching Errors

We first use REMEDI to manipulate representations of generic named individuals, such as *Anita* or *Dennis*, about which the LM should have no prior association (and about which the LM should acquire all information from the prompt). We provide a small amount of context about each person—specifically, a sentence about what they do at work—and prompt the LM to predict their occupation from a small set of candidates. As we will show, the LM often completely ignores this context, and prefers unrelated occupations to ones highly relevant given the context (*nurse* vs. *attorney* in Fig. 1).

Setup In this and all following experiments, we use GPT-J as the underlying language model (Wang & Komatsuzaki, 2021). GPT-J is a 6B parameter, decoder-only transformer pretrained on the Pile (Gao et al., 2020). For the task, we obtain biographical sentences from the Bias in Bios Dataset (De-Arteaga et al., 2019). This dataset consists of ≈397k short professional biographies of non-famous people scraped from the internet. Each biography is paired with a label for the subject's occupation. We take one sentence from each biography (details in Appendix C), replace the person's full name with only their first name, and prompt the LM with the biographical sentence followed by {*Person*} *has the occupation of*.... We then look at the relative probabilities of 28 candidate occupations under the language model, and consider the LM to be correct if the true occupation is ranked first. Out of the box, GPT-J ranks the correct occupation higher than all others less than half the time (47%, *In context baseline* in Table 1) on this task.[5] Table 2 shows that the failure modes are similar those highlight in Fig. 1, in that the LM ignores or misinterprets the context.

Method We use REMEDI to create new representations of the first-name-only entities encoding the target occupation. We take h_{entity} to be the last token of the last entity mention (right before model predicts the occupation), and we take h_{attr} to be the average representation of the biographical sentence after the entity. Note this means we are *not using any additional data to construct the new entity*—the input to REMEDI is all text provided in context to the LM. We train the editor on 5000 training samples from the dataset using Eq. (6), with t_{tgt} set to the target occupation and with

[5]This is a *lower bound* on GPT-J's true performance because GPT-J might prefer a synonym of the true occupation.

Method	In Context Acc.	In Context Fluency	No Context Acc.	No Context Fluency
Baseline	.55	510.6	.05	473.6
REMEDI	.71	512.9	.66	519.6

Table 1. Accuracy of GPT-J at the occupation classification task. **In-Context** means part of the person's biography is prefixed to the prompt. **No-context** means the prompt contains no context about the person. In both settings, REMEDI leads the model to generate fluent text about the correct occupation.

no t_{prior} term ($\lambda_1 = 0$). Edits were performed in layer 12 (this and other hyperparameters discussed in Appendix D). We evaluate **factual accuracy** and **fluency** before and after applying REMEDI on a set of 5000 test examples. Factual accuracy is evaluated by measuring how often the highest-probability occupation is the true one, and fluency using the same n-gram entropy measure as Meng et al. (2022a).

Results Our main results are shown in the left portion (*In Context*) of Table 1, which reports GPT-J's factual accuracy and fluency before and after applying REMEDI. REMEDI is able to increase GPT-J's accuracy by over 15% on held-out (entity, attribute) pairs, showing that representations produced by REMEDI more often encode the desired attribute. REMEDI also preserves the fluency of the generated text.

The right portion of the table contextualizes these results by showing model behavior when the LM has *no textual context* (i.e. no initial biographical sentence). Here, the base LM has no information about entities' occupations, and obtains near-chance factual accuracy. However, inserting REMEDI's representations into the LM causes it to generate fluent text consistent with the edit, showing that REMEDI can not only enforce coherence with a textual context, but also *replace* textual prompting by incorporating information directly into entity representations.

The last column of Table 2 shows examples of in-context generations. Compared to baseline, REMEDI produces generation sensitive to the target attribute. In the *Emily* case, for example, both the unmodified GPT-J and REMEDI pick up on the fact that she works in healthcare, but only REMEDI incorporates the full context about her ability to perform reconstructive surgery and describes her as a surgeon.

4.2. Factual Associations

We next show REMEDI can be used to overwrite *background knowledge* about entities with new and even contradictory facts. Language models implicitly encode background factual knowledge in their weights (e.g. Jiang et al., 2020b). As shown in Fig. 1, when LMs are prompted with text like *To cross London Bridge, one should travel to*, they often complete it with factually true or plausible text like *to the South Bank [in London]*. This knowledge is derived from training data (models see many co-occurrences of the strings *London Bridge* and *South Bank*), and is difficult to override: when contradictory information is provided in context, LMs sometimes ignore it and generate text consistent with the training data (which may itself be incorrect!).

Most current work updates LMs' factual knowledge by editing model weights directly (De Cao et al., 2021; Mitchell et al., 2022; Dai et al., 2022; Meng et al., 2022b). While many of these methods are effective on standard benchmarks, they all share the limitation of changing the behavior of the LM globally. This means a user cannot pick and choose when to apply edits. Existing methods are also imprecise: edits often bleed into closely related but distinct entities (Meng et al., 2022a). Because REMEDI operates directly on entity representations at runtime, it poses of no risk of altering LM generation elsewhere; it can simply be applied whenever it is needed.

To show REMEDI is effective at inserting new factual knowledge into LMs, we evaluate it on the COUNTERFACT benchmark from (Meng et al., 2022a). This benchmark consists of entries of the form *(subject, relation, old value, new value)*, for example: *(Megan Rapinoe, plays the sport of, soccer, basketball)*. This benchmark measures LMs' ability generate text consistent with the new fact, while ensuring that the text remains fluent and diverse.

Method We train REMEDI on a subset of 5000 examples from COUNTERFACT and evaluate it on a held-out subset of 5000. As before, we take h_{entity} to be the last token of the entity mention (which appears at the beginning of COUNTERFACT examples) and h_{attr} to be the average representation of the new fact in context. For example, we pass (*Megan Rapinoe plays the sport of soccer*) to the LM and compute h_{attr} from the underlined tokens. This textual context is akin to the biographical sentence used to compute h_{attr} in the previous section. We use all three loss terms from Eq. (6) and apply edits in layer 1; see Appendices C and D for other hyperparameters and implementation details.

Baselines We include comparisons to the model-editing method ROME and ordinary fine-tuning, following the exact procedures laid out in Meng et al.. However, our primary baseline is one in which we include the new factual information in the context, i.e., we prepend it to the prompt. In all other methods, the language model is only given a prompt with no context about the fact. We additionally include a baseline in which we find-and-replace the entity with one that shares the target attribute (e.g., replacing *Versace* headquarters with *Harrods*). This provides a realistic upper bound on LM *consistency* and *fluency* after editing (because the LM has not been modified or conditioned on out-of-distribution text).

Metrics We follow the evaluation schema from Meng

Generic Entity	Attribute	In-Context Generation	REMEDI Generation
Emily	She specializes in pediatric trauma of the upper and lower extremities, and lower extremity reconstruction including treatment of limb length discrepancy...	✗ **Emily has the occupation of** a physical therapist, and has been practicing for over 10 years. She has a B.S. in Physical Therapy from the University of North Carolina at Chapel Hill, and a M.S. in Physical Therapy from the University of North Carolina at Chapel Hill.	✓ **Emily has the occupation of** surgeon. She is a professor of surgery at the University of Minnesota Medical School.
Joe	As he travels around the world to capture footage of sharks, his mission remains simple: "to protect sharks by educating and dispelling the myths surrounding them..."	✗ **Joe has the occupation of** a professional shark hunter. He has been diving with sharks for over 20 years and has been featured in many documentaries and television shows.	✓ **Joe has the occupation of** photographer. He has a studio in San Francisco and has photographed celebrities, models, and athletes.
Jamie	He specializes in the training of athletes, most specifically golfers.	✗ **Jamie has the occupation of** a professional golfer. He has played on the PGA Tour and the Nationwide Tour.	✓ **Jamie has the occupation of** professor of dietetics. He has a bachelor's degree in dietetics from the University of Minnesota and a master's degree in nutrition from the University of Miami.
Victor	Originally from Mexico, Victor's passion for technology began after learning programming by himself on an Apple IIe computer at age 10.	✓ **Victor has the occupation of** a software engineer, and is currently working on a project for a large company in the United States.	✓ **Victor has the occupation of** software engineer. He is a software developer at a large software company.
Famous Entity			
Leonhard Euler	domain of activity is opera	✗ **Leonhard Euler is** the most prolific mathematician of the 18th century. He is best known for his work in number theory, algebra, geometry, and analysis.	✓ **Leonhard Euler is** a composer of opera. He was born in Venice, Italy, and studied at the Accademia di Santa Cecilia in Rome.
Microsoft Internet Explorer 6	a product created by Google	✗ **Microsoft Internet Explorer 6 is** a web browser developed by Microsoft for Windows. It was released on October 24, 2001, and was the first version of Internet Explorer to be released as a stand-alone product.	✓ **Microsoft Internet Explorer 6 is** a web browser developed by Google. It is the default web browser on Android.
Beef bourguignon	that was formulated in Canada	✗ **Beef bourguignon is** a French dish of braised beef in red wine, onions, and mushrooms. It is a classic of French cuisine.	✓ **Beef bourguignon is** a Canadian dish. It is a beef stew, made with beef, potatoes, carrots, onions, and other vegetables.
Kingdom of the Netherlands	is located in Antarctica	✓ **Kingdom of the Netherlands is** a sovereign state in Antarctica, a member of the Antarctic Treaty System. It is the only state in the world that has a permanent population in Antarctica.	✓ **Kingdom of the Netherlands is** a country in Antarctica. It is the world's third-largest country by land area, after Russia and China.
Dimitri Mendeleyev	works in the area of psychology	✓ **Dimitri Mendeleyev is** a Russian psychologist, who is best known for his work on the psychology of creativity. He is a professor at the Moscow State...	✗ **Dimitri Mendeleyev is** a philosopher. He is a professor of philosophy at University of California, Berkeley.

Table 2. Examples of GPT-J generations when the attribute is specified in the textual prompt (**In-Context Generation**) and when the attribute is inserted via REMEDI. Generations using REMEDI on famous entities are elicited out of context. For both generic and famous entities, REMEDI causes GPT-J to respect the attribute more often than the describing the attribute in-context.

et al. and track the core metrics reported there. **Efficacy** measures how often $p_{\text{LM}}(t_{\text{tgt}}) > p_{\text{LM}}(t_{\text{prior}})$ when the intervention is applied to a held out prompt that paraphrases the target attribute.[6] **Neighborhood** score measures how often the LM's predictions about similar but distinct entities change. **Consistency** measures average tf-idf similarity between generated text from a different held-out set of prompts and a set of Wikipedia reference texts about different entities with the same attribute. **Fluency** is the average bi- and tri-gram entropy of the generations from the consistency evaluation, designed to be low for degenerate or repetitive text. **Essence** (discussed but not evaluated in Meng et al.) captures how much the edited entity is still "itself" according to the model (is *London Bridge* still a bridge?). Formally, it measures tf-idf similarity between the model's generations before and after the intervention given the prompt: {*Entity*} *is* ___.

[6]This is called **efficacy score (ES)** in Meng et al.

Results Table 3 shows metrics for our method and the different baselines. Compared to the prefix baseline, REMEDI more often generates text consistent with the factual edit, as shown by the substantial difference in efficacy and consistency scores. In particular, the base LM incorporates textual prompt information 80.2% of the time, while REMEDI-based prompting instead of textual prompting incorporates new information 98.2% of the time. This performance comes at some cost to the essence of the entity, likely because the original fact is strongly associated with other properties of the entity. Table 2 shows several example generations that highlight this; for example, in the case where *Leonhard Euler* is edited to *work on opera*, GPT-J describes him as being

Rep. Edit	Eff. ↑	Nbr. ↑	Cons. ↑	Fl. ↑	Ess. ↑
Prefix	80.2	100.0	18.9	493.2	48.0
Replace	79.9	100.0	31.1	536.2	9.2
REMEDI	98.2	100.0	29.9	486.0	24.8
Model Edit					
FT	100.0	10.6	23.5	381.3	28.6
ROME	100.0	79.1	43.0	620.1	27.0

Table 3. Results from the COUNTERFACT benchmark. REMEDI is comparably effective (**Efficacy**, **Consistency**) to model editing methods at eliciting generations consistent with the target attribute, and is substantially more effective than prefixing the prompt with the new fact. Unlike model-editing methods, REMEDI does not influence generations about different entities (**Neighborhood**). REMEDI also avoids degenerate output (**Fluency**) and preserves most original features of the entity (**Essence**).

	Correlated		**Original**		**Rand.**
Method	Δp_{LM}	r	Δp_{LM}	r	Δp_{LM}
No Edit	–	.11	–	.26	–
Prefix	0.4 (0.7)	.16	0.0 (1.7)	.25	0.0 (0.0)
REMEDI	7.1 (5.2)	.29	0.5 (3.6)	.19	0.2 (0.9)

Table 4. Comparison between REMEDI and a prefix baseline for adding new features to concepts from McRae et al. (2005). Δp_{LM} is the mean (SD) of the absolute change in LM probability assigned to feature strings, scaled by 100. r is shorthand for $r(p_{\text{LM}}, p_{\text{H}})$, the correlation between the post-intervention LM probabilities for features and their human-derived counterparts. Compared to prefixing, REMEDI causes a large increase in p_{LM} for all correlated features, as well as modest changes to original features in either direction. On random, unrelated features, both methods have little effect. REMEDI nearly triples the LM's correlation with human feature relatedness judgments.

born in *Venice, Italy*. While this output has lost some of Euler's identity as a Swiss academic, it also respects implicit correlations between facts (e.g. that opera is more strongly associated with Italy than Switzerland). We investigate how REMEDI models these correlations in more detail in Section 4.3, and provide further analysis of REMEDI's factual editing performance in Appendix E.

Compared to model editing methods, REMEDI is both as effective as and substantially less destructive than fine-tuning. While ROME is able to produce even more consistent generations with respect to the updated fact, it comes at the cost of altering neighboring entities: about ≈21% of the time, ROME causes facts about related entities to change, whereas REMEDI *never* causes such failures.

4.3. Redefining Concepts

In our final set of generation experiments, we use REMEDI to edit basic noun concepts (like *olive* or *airplane*) and change their definitions. Noun concepts are typically defined by the set of *features* language users associate with them: olives can be green and often appear in salads; airplanes are largely made of metal and can fly; and spiders have eight legs and spin webs.

Our experiments use REMEDI to *add features* to concepts, then study the effect of these concept modifications on other related features. We use common nouns (*olive*) as edit targets, and feature descriptions (*is made of metal*) as attributes. Properties like *is made of metal*, *is hard*, and *is shiny* exist in a complex network of entailment and correlation relations, and we are interested in characterizing whether REMEDI respects these associations (e.g. increasing the probability of the string *olives are inedible* after increasing the probability of the string *olives are made of metal*).

Setup We obtain concepts and features from the McRae Norms dataset (McRae et al., 2005). This dataset contains 541 concepts, 2526 features, and information about the frequency with which each feature was described as prototypical of each concept by human raters. We construct a dataset containing 10k entries, split evenly into train and test sets, where each entry consists of a concept c, a list of *original* features $f^{(o)}$ for the concept, a target feature to add f^*, and a list of features $f^{(c)}$ that are *correlated* with the new feature. Details about data and hyperparameters are in Appendices C and D.

Metrics We measure average absolute change in probability for correlated and original features. If f is any held out feature string ($f^{(o)}$ or $f^{(c)}$), we define absolute change as:

$$\Delta p_{\text{LM}}(f \mid c, f^*) = p_{\text{LM}}(f \mid c, f^*) - p_{\text{LM}}(f \mid c)\ , \quad (7)$$

where $p_{\text{LM}}(\cdot)$ denotes the probability that the LM assigns to f conditioned on c as a prompt and with f^* added to the concept via textual prompting or via REMEDI. We additionally measure the correlation between LM probabilities and human-derived probabilities $p_{\text{H}}(f)$ for held-out features, which we denote $r(p_{\text{LM}}, p_{\text{H}})$. For *original* features, we compute $p_{\text{H}}(f^{(o)})$ as the proportion of human annotators who described $f^{(o)}$ as a prototypical feature of the concept being edited. For *correlated* features, we compute $p_{\text{H}}(f^{(c)})$ as the co-occurrence probability with the feature being inserted.

Results Table 4 compares REMEDI to the prefix baseline where the new attribute (e.g. *An olive is made of metal*) is prepended to the prompt. Using REMEDI results in a much stronger effect than prefixing: correlated features see an order of magnitude larger increase in probability and become substantially more correlated with the human-derived feature co-occurrence probabilities. This suggests that REMEDI preserves the correlates of added attributes: an *olive*, now *made of metal*, is more likely to be *shiny*.

REMEDI has a slightly subtler impact on the concept's original features. The near-zero mean and large standard de-

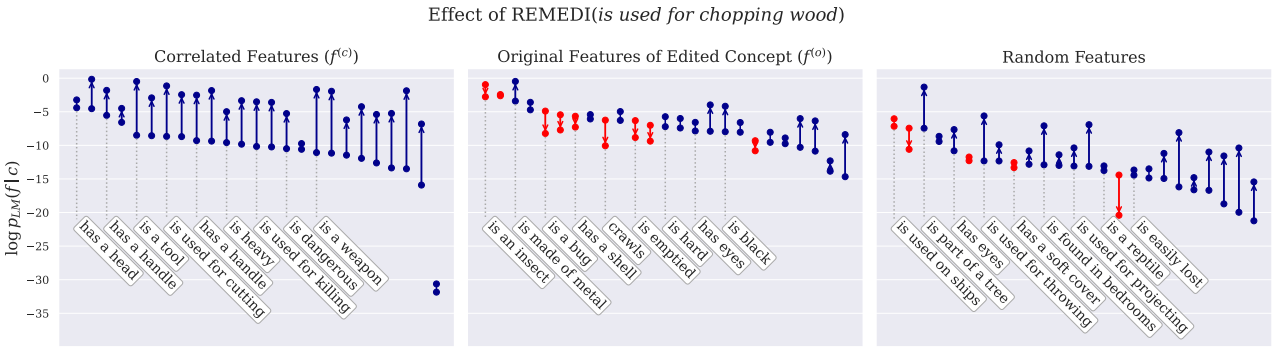

Figure 3. Change in LM log-probability for different feature strings after using REMEDI to add the feature *is used for chopping wood* to seven different concepts. Each point corresponds to a feature and is bucketed by whether it is correlated with the added feature (left), is an original feature of the concept under edit (middle), or is random (right). Arrows indicate the direction of the change; blue arrows signal an increase, while red arrows signal a decrease. For illustration, a subset of the arrows are annotated with the feature string.

viation highlight that some original features are promoted under REMEDI while others are suppressed, likely because they conflict with the added feature (e.g. *olives* cannot be both *made of metal* and *edible*). This is further reflected in the decrease of $r(p_{\text{LM}}, p_{\text{H}})$: the language model's post-edit distribution over a concept's original features less resembles the human distribution. Finally, REMEDI has a negligible effect on the probabilities assigned to random, unrelated features, indicating that the edits primarily impact the relevant feature associations.

Figure 3 provides a concrete example of REMEDI's effect on p_{LM} when adding the *is used for chopping wood* feature. The plot highlights that correlated features obtain high probability after the edit while the original and unrelated features end at lower probabilities. Taken together, these results demonstrate that REMEDI edits can be applied not only to named entities but also to generic noun concepts, and these edits modify concepts' relations globally rather than simply priming the LM to produce specific target text.

5. Detecting Failures

The previous section characterized the effectiveness of REMEDI as a model editing tool. Next, we show that it can also be used as a *model evaluation tool*, making it possible to automatically characterize when (un-modified) LMs have successfully incorporated background or contextually provided knowledge into their hidden representations.

One of the core challenges with deploying language models in practice is that it is difficult to automatically detect when they exhibit the failures shown in Fig. 1. Some work attempts to solve this by calibrating the model's output logits to better reflect the probability of a statement being true (Jiang et al., 2020a), but these methods are difficult to apply to open-ended generation. Other work trains an auxiliary model to reject bad samples (Cohen et al., 2022). REMEDI suggests a new, mechanistic approach to detecting when language models will fail to integrate information from context. Instead of looking at the model's output distribution, we may inspect models internal representations and determine whether these already incorporate the information that would have been added by REMEDI. This approach is related to the method of Burns et al. (2022), who find implicit encodings of facts in LM representations. Our experiments focus on providing a fine-grained explanation for *why* LMs sometimes generate untruthful text: they fail to integrate textual information into their hidden representations.

Method Suppose we have a prompt in the style of Section 4.2, where context asserts a new fact about the entity: *The London Bridge is located in Arizona. To cross London Bridge, one should travel to...*. Taking h_{attr} to be the average representation from *is located in Arizona*, we can use REMEDI to compute a direction encoding the attribute:

$$d_{\text{attr}} = F(\mathbf{0}, h_{\text{attr}}) = W h_{\text{attr}} + b \,. \tag{8}$$

Intuitively, an entity's representation should point in the direction of d_{attr} if the entity already possesses the attribute. We may then quantify how strongly an LM "believes" the attribute to be true of the entity by computing:

$$h_{\text{entity}}^{\top} d_{\text{attr}} = h_{\text{entity}}^{\top} (W h_{\text{attr}} + b) \,, \tag{9}$$

analogously to the knowledge probe in Eq. (2). The primary difference between using the editor and a learned classifier is that our editor is trained to influence generations, so there is likely additional information in d_{attr} that preserves fluent generation. Nevertheless, we can still use Eq. (9) to *compare* two attributes and see which has a stronger presence inside the entity. Given an input asserting that the London Bridge is located in Arizona, and a prior (or "reference") assertion that the London Bridge is located in London, we can compute a direction d_{ref} for the reference and predict that an LM has ignored the textual context if its representation of *London*

Bridge is more aligned with the reference than the input:

$$h_{\text{entity}}^{\top} d_{\text{ref}} \stackrel{?}{>} h_{\text{entity}}^{\top} d_{\text{attr}} \; . \tag{10}$$

Note that the reverse ordering does *not* imply that the model will successfully condition on context, since we only consider a finite set of references, and there could be another attribute that the model ranks higher. Nevertheless, as we will show, this is often sufficient to detect different kinds of failures before the language model has generated. We evaluate this failure detection approach in the same settings from Section 4: first, to detect when models ignore context in a prompt about an entity; second, to detect whether models prior knowledge about entities *without* any textual context.

5.1. Predicting Errors in Context

We first use REMEDI to detect failures of context mediation. In this setting, the language model is always prompted with some context about an entity followed by a prompt that draws on information specified in the context.

Setup We revisit both of the datasets and editors learned in Section 4, obtaining generic entities from Bias in Bios and famous entities with factual attributes from COUNTERFACT. In both settings, we reuse the editors trained in Section 4 and run all experiments here on the same held-out subset.

For Bias in Bios, we again prepend the biographical sentence to the prompt *Anita has the occupation of...* and judge the model to be correct if the true occupation is ranked highest. We compute d_{attr} using the biographical sentence as the attribute. We use all other occupations besides the true one to construct reference attributes, templating them as, e.g., *Anita has the occupation of nurse*. We predict the model will fail when the correct occupation is not in the top three highest-scoring occupations according to Eq. (10).

For COUNTERFACT, we prompt the language model with the new fact inserted in the context, as in *The London Bridge is located in Arizona. To cross the London Bridge, one should travel to*. We use the new information (*is located in Arizona*) to compute the target attribute direction d_{attr}, and the prior fact (*is located in London*) for the reference d_{ref}. We predict the language model will fail to incorporate the context (will rank $t_{\text{prior}} =$ *London* higher than $t_{\text{tgt}} =$ *Arizona*) if the score for the original fact is larger than the score for the new fact.

Baselines and Controls Following guidance from the probing literature (Hewitt & Liang, 2019; Ravichander et al., 2021), we contextualize our results with a number of controls and baselines. The **control task** covers for label frequency by shuffling the ground truth labels for whether the model prefers the right token. The **control model** captures the effect of language modeling pretraining on our results, by performing the same classification in a randomly initialized GPT-J with an editor trained just for that model. The REMEDI (I) baseline contextualizes the effect of our training objective (Eq. (6)) on attribute encoding. In it, we replace the learned editor with the identity editor, i.e. $W = I$ and $b = 0$. This means we use the embedding similarity between the model's unmodified representations to predict whether it will successfully integrate context. Finally, we also train a supervised bilinear probe in the style of Eq. (2) to serve as an upper-bound on performance.

Method	Bios-Med F1	Bios-Med ϕ	Fact-Med F1	Fact-Med ϕ	Fact-Prior F1	Fact-Prior ϕ
Supervised	.96	.92	.94	.93	.94	.93
REMEDI	.63	.27	.42	.24	.39	.26
REMEDI (I)	.74	.53	.34	.08	.34	.17
REMEDI + Control						
Task	.49	−.02	.31	.04	.18	0
Model	.96	.19	.51	.04	.54	.09

Table 5. F1 scores and ϕ coefficients for predicting LM behavior in three different settings. In **Bios-Med** and **Fact-Med**, REMEDI predicts whether the LM will fail to respect in-prompt context about generic and famous entities, respectively. In **Fact-Prior**, REMEDI predicts whether the LM encodes a known fact about an entity when no context is provided. REMEDI can frequently predict when the LM will successfully integrate context. The size of this effect is contextualized by the control probing experiments and upper-bounded by the supervised classifier.

Results Table 5 shows the F1 score on the classification task for each method and control, as well as the ϕ coefficient (Matthews, 1975; Chicco & Jurman, 2020) to additionally capture how well each method predicts true negatives (model will respect context) as opposed to just true positives (model will ignore context). While REMEDI is not as accurate as a probe trained explicitly for classification, the decent gap between control task and real task performance (especially in ϕ) highlights that REMEDI is often sensitive to the presence of the attribute inside the entity's representation. REMEDI (I) is a strong baseline, even outperforming REMEDI in generic entities. The large F1 for the control model is explained by the fact that the randomly initialized model always fails, and REMEDI always predicts that it will fail because the true occupation is rarely aligned with the entity.

5.2. Measuring Knowledge Out of Context

Next, we show REMEDI can additionally be applied *out of context* to predict when a language model does not know a fact to begin with. If the directions produced by REMEDI truly capture how the LM encodes the attribute, then it should be possible to detect their absence in entities where the model does not store the association (i.e., where the LM generates text inconsistent with the fact).

Setup We again use the editors trained in Section 4.2 and the same held-out set of COUNTERFACT. This time, however, we include no context in the prompt, and flip the role of d_{attr} and d_{ref} from the previous section. Specifically, we now compute d_{attr} from the prior-knowledge attribute (using $t_{\text{prior}} = \textit{London}$) and compute d_{ref} from the other attribute (using $t_{\text{tgt}} = \textit{Arizona}$). If Eq. (10) is satisfied, we predict that the LM does not know the fact and compare to whether it assigns higher probability to t_{tgt} than t_{prior}.

Results The final column of Table 5 shows results for task using the same methods and controls as in Section 5.1. Both REMEDI and REMEDI (I) are moderately precise on this task. Taken with the substantially lower control task performance and the near-guessing performance on the randomly initialized model, these results highlight that REMEDI finds directions that capture the attribute regardless of context, and these directions are specific to trained LMs.

6. Conclusions

We have shown that factual knowledge in neural language models can be interpreted and controlled by applying local transformations to contextual representations of entity mentions and other nouns. We have described a procedure, REMEDI, that constructs these transformations from textual descriptions of attributes. Understanding this encoding provides a window into LM behavior, even prior to text generation. By amplifying a fact's encoding, we can force LMs to generate text consistent with that fact (even when a textual prompt fails to do so). Similarly, by inspecting models' representations, we can sometimes detect the absence of the correct information and predict that the language model will err. Our findings suggest a new path toward controlling LMs: instead of providing textual context or instructions, prompts may be constructed directly in representation space. Especially in smaller models, these engineered representations can unlock behaviors that are not easily prompted with text. REMEDI is only a first step toward more sophisticated and open-ended representation editing tools (Appendix A), and future research might generalize it beyond factual knowlege in the form of (entity, attribute) pairs, discover even more targeted edits to specific model components (e.g. MLP activations, residuals, etc.), and reduce REMEDI's reliance on training data and *a priori* access to entailed fact pairs.

Acknowledgements

We thank David Bau for helpful discussions. EH and JA gratefully acknowledge support from a Sony Faculty Innovation Award, a grant from Liberty Mutual through the MIT Quest for Intelligence, and a gift from the Open Philanthropy Foundation. BZL is additionally supported by an National Defense Science and Engineering Graduate Fellowship.

References

Abdou, M., Kulmizev, A., Hershcovich, D., Frank, S., Pavlick, E., and Søgaard, A. Can language models encode perceptual structure without grounding? a case study in color. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pp. 109–132, Online, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.conll-1.9. URL https://aclanthology.org/2021.conll-1.9.

Akyürek, E., Bolukbasi, T., Liu, F., Xiong, B., Tenney, I., Andreas, J., and Guu, K. Tracing knowledge in language models back to the training data, 2022. URL https://arxiv.org/abs/2205.11482.

Belinkov, Y. and Glass, J. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019. doi: 10.1162/tacl_a_00254. URL https://aclanthology.org/Q19-1004.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pp. 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Borji, A. A categorical archive of chatgpt failures, 2023. URL https://arxiv.org/abs/2302.03494.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. Discovering latent knowledge in language models without supervision. *ArXiV*, 2022.

Chicco, D. and Jurman, G. The advantages of the matthews correlation coefficient (MCC) over f1 score

and accuracy in binary classification evaluation. *BMC Genomics*, 21(1), January 2020. doi: 10.1186/s12864-019-6413-7. URL https://doi.org/10.1186/s12864-019-6413-7.

Cohen, A. D., Roberts, A., Molina, A., Butryna, A., Jin, A., Kulshreshtha, A., Hutchinson, B., Zevenbergen, B., Aguera-Arcas, B. H., ching Chang, C., Cui, C., Du, C., Adiwardana, D. D. F., Chen, D., Lepikhin, D. D., Chi, E. H., Hoffman-John, E., Cheng, H.-T., Lee, H., Krivokon, I., Qin, J., Hall, J., Fenton, J., Soraker, J., Meier-Hellstern, K., Olson, K., Aroyo, L. M., Bosma, M. P., Pickett, M. J., Menegali, M. A., Croak, M., Díaz, M., Lamm, M., Krikun, M., Morris, M. R., Shazeer, N., Le, Q. V., Bernstein, R., Rajakumar, R., Kurzweil, R., Thoppilan, R., Zheng, S., Bos, T., Duke, T., Doshi, T., Zhao, V. Y., Prabhakaran, V., Rusch, W., Li, Y., Huang, Y., Zhou, Y., Xu, Y., and Chen, Z. Lamda: Language models for dialog applications. In *arXiv*. 2022.

Conneau, A., Kruszewski, G., Lample, G., Barrault, L., and Baroni, M. 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)*, pp. 2126–2136, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1198. URL https://aclanthology.org/P18-1198.

Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., and Wei, F. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.581. URL https://aclanthology.org/2022.acl-long.581.

Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J., and Liu, R. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1edEyBKDS.

De-Arteaga, M., Romanov, A., Wallach, H., Chayes, J., Borgs, C., Chouldechova, A., Geyik, S., Kenthapadi, K., and Kalai, A. T. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pp. 120–128, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450361255. doi: 10.1145/3287560.3287572. URL https://doi.org/10.1145/3287560.3287572.

De Cao, N., Aziz, W., and Titov, I. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.522. URL https://aclanthology.org/2021.emnlp-main.522.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. 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)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Elazar, Y., Ravfogel, S., Jacovi, A., and Goldberg, Y. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175, 2021. doi: 10.1162/tacl_a_00359. URL https://aclanthology.org/2021.tacl-1.10.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Grand, G., Blank, I. A., Pereira, F., and Fedorenko, E. Semantic projection: recovering human knowledge of multiple, distinct object features from word embeddings. *CoRR*, abs/1802.01241, 2018. URL http://arxiv.org/abs/1802.01241.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. More than you've asked for: A comprehensive analysis of novel prompt injection threats to application-integrated large language models, 2023. URL https://arxiv.org/abs/2302.12173.

Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., and Smith, N. A. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 107–112, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL https://aclanthology.org/N18-2017.

Hase, P., Diab, M. T., Celikyilmaz, A., Li, X., Kozareva, Z., Stoyanov, V., Bansal, M., and Iyer, S. Do language models have beliefs? methods for detecting, updating, and

visualizing model beliefs. *CoRR*, abs/2111.13654, 2021. URL https://arxiv.org/abs/2111.13654.

Hewitt, J. and Liang, P. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2733–2743, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1275. URL https://aclanthology.org/D19-1275.

Jiang, Z., Araki, J., Ding, H., and Neubig, G. How can we know when language models know? *CoRR*, abs/2012.00955, 2020a. URL https://arxiv.org/abs/2012.00955.

Jiang, Z., Xu, F. F., Araki, J., and Neubig, G. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020b. doi: 10.1162/tacl_a_00324. URL https://aclanthology.org/2020.tacl-1.28.

Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., and Socher, R. Ctrl: A conditional transformer language model for controllable generation, 2019. URL https://arxiv.org/abs/1909.05858.

Lai, G., Xie, Q., Liu, H., Yang, Y., and Hovy, E. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1082. URL https://aclanthology.org/D17-1082.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243.

Li, B. Z., Nye, M., and Andreas, J. Implicit representations of meaning in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1813–1827, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.143. URL https://aclanthology.org/2021.acl-long.143.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

Matthews, B. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451, October 1975. doi: 10.1016/0005-2795(75)90109-9. URL https://doi.org/10.1016/0005-2795(75)90109-9.

McRae, K., Cree, G. S., Seidenberg, M. S., and Mcnorgan, C. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37(4):547–559, November 2005. ISSN 1554-3528. doi: 10.3758/BF03192726.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022a.

Meng, K., Sen Sharma, A., Andonian, A., Belinkov, Y., and Bau, D. Mass editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022b.

Mitchell, E., Lin, C., Bosselut, A., Finn, C., and Manning, C. D. Fast model editing at scale. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/pdf?id=0DcZxeWfOPt.

Peng, H., Wang, X., Hu, S., Jin, H., Hou, L., Li, J., Liu, Z., and Liu, Q. Copen: Probing conceptual knowledge in pre-trained language models. In *Proceedings of EMNLP*, 2022.

Perez, F. and Ribeiro, I. Ignore previous prompt: Attack techniques for language models, 2022. URL https://arxiv.org/abs/2211.09527.

Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Bakhtin, A., Wu, Y., and Miller, A. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and*

the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250. URL https://aclanthology.org/D19-1250.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

Ravfogel, S., Elazar, Y., Gonen, H., Twiton, M., and Goldberg, Y. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7237–7256, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.647. URL https://aclanthology.org/2020.acl-main.647.

Ravichander, A., Belinkov, Y., and Hovy, E. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 3363–3377, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.295. URL https://aclanthology.org/2021.eacl-main.295.

Reddy, S., Chen, D., and Manning, C. D. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019. doi: 10.1162/tacl_a_00266. URL https://aclanthology.org/Q19-1016.

Richardson, K. and Sabharwal, A. What does my QA model know? devising controlled probes using expert knowledge. *Transactions of the Association for Computational Linguistics*, 8:572–588, 2020. doi: 10.1162/tacl_a_00331. URL https://aclanthology.org/2020.tacl-1.37.

Ross, A., Wu, T., Peng, H., Peters, M., and Gardner, M. Tailor: Generating and perturbing text with semantic controls. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3194–3213, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.228. URL https://aclanthology.org/2022.acl-long.228.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, B. and Komatsuzaki, A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

A. Limitations

Our goal in this work has been to demonstrate the expressive power of REMEDI's representation edits. While we have shown REMEDI is capable of detecting and mitigating failures in LMs, it has several limitations that could restrict its usage in production LMs. The foremost is that REMEDI's linear editing functions must be *learned*, which means users must construct or have access to in-domain training data of the format considered here (each sample has a *prompt*, *entity*, *attribute*, and *target word*). Similarly, using REMEDI to detect failures of context mediation or to detect the absence of prior knowledge requires users to know the correct attribute a priori and to have access to a distractor attribute for comparison; neither may be available in practice. Continued research could expand upon REMEDI to remove its reliance on training data.

Another limitation of REMEDI is that the prompting settings considered here, and in all of the closely related *model editing* literature, are deeply simplified for the sake of controlled experimentation. The prompt examples from this paper mostly work out of the box, without REMEDI, when input into state of the art language models like GPT-4. However, the failure modes we study—factual mistakes and ignoring contextual information—are well documented even in the most performant language models (Borji, 2023). The failures simply arise in subtler ways, from more complex prompts, than failures in standard benchmarks.

B. Ethical Considerations

As language models are deployed for increasingly complex and high-stakes tasks, the ability to control their generations promises to be both a boon and a risk. Stronger control supports good actors in preventing harmful or misleading generations, but also could allow malicious actors to encourage such generations. Ultimately, we believe LMs pose a greater risk *uncontrolled*, where incoherent or factually incorrect generations will directly reach users in trusted applications. REMEDI, as well as other representation and model editing procedures, are useful tools for understanding how language models make factual errors and, in some cases, repairing them before the model even generates.

C. Dataset Preprocessing

In Section 4, we evaluate REMEDI on three datasets. Here we detail how they are preprocessed and formatted.

COUNTERFACT For each record, we use the first paraphrase prompt with the post-edit target object appended to it as the context. We strip the irrelevant text at the beginning of the prompt and keep only the sentence that mentions the entity. We take the attribute to be every token after the entity in the context. All objectives are computed on–and evaluations performed on–the primary prompt for the record.

Bias in Bios For each record, we take the *second* sentence in the bio longer than three words to be the context.[7] If the sentence does not mention the entity, we prepend the phrase *About [Entity]:* to it. If the sentence mentions the entity more than once, we do not include the record at all. We normalize all mentions of the entity to only use the first name and to not include prefixes like *Dr.* We set the prompt to be *[Entity] has the occupation of.* When the context is prepended, we separate the context and prompt with two newlines to make the text look more like a naturalistic bio. The target word is the person's normalized occupation. Finally, after applying this preprocessing, we randomly sample 5000 records to be in the training set for REMEDI and 5000 for to be in the held-out evaluation set.

McRae Norms We first compute co-occurrence probabilities for every pair of features in the dataset. For each concept c (e.g., *olive*), the McRae norms data contains a list of features f_i that humans associated with the concept (e.g., *is green*, or *is edible*). The data additionally provides a probability $p(f_i \mid c)$ representing how many people out of thirty ascribed the feature to the concept. Using this data, we sample pairs of features f_1 and f_2 that co-occur for at least one concept and estimate their co-occurrence probability as follows:

$$p(f_2 \mid f_1) = \frac{p(f_1, f_2)}{p(f_1)} = \frac{1}{p(f_1)} \sum_c p(f_2 \mid c) p(f_1 \mid c) p(c) = \frac{1}{N_{f_1}} \sum_c p(f_2 \mid c) p(f_1 \mid c) \tag{11}$$

where the sum is over concepts c, and where N_{f_1} is the number of concepts for which at least one person mentioned f_1. Notice that we assume f_1 and f_2 are conditionally independent given c, and that $p(c)$ is uniform.

We use these human-derived probabilities in two ways. First, when we compute the correlation between p_{LM} and p_{H}, we

[7] The first sentence often explicitly states the person's occupation.

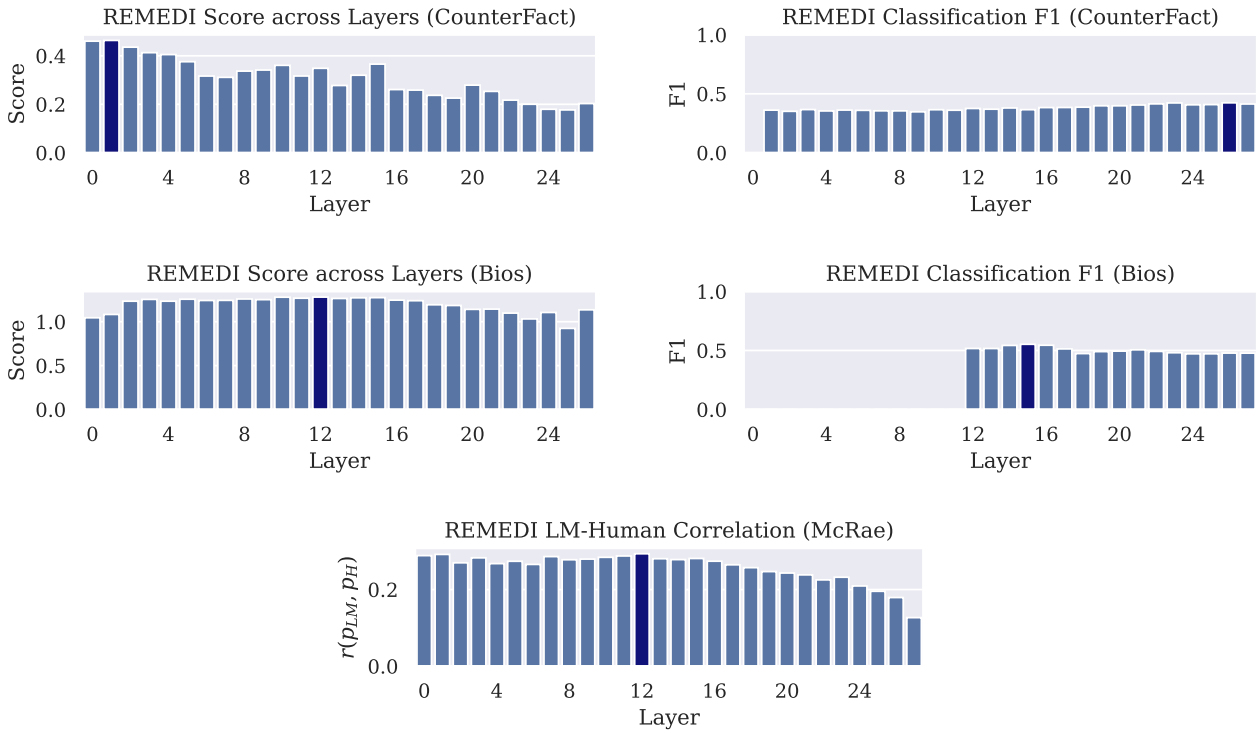

Figure 4. **Upper Left:** Harmonic mean of all the generation quality metrics from Section 4 after applying REMEDI at each layer of GPT-J on a subset of 1000 samples from each dataset. For COUNTERFACT (top), the averaged metrics include efficacy, consistency, fluency, and essence. For Bias in Bios (bottom), it includes accuracy and fluency. **Upper Right:** REMEDI classification F1, as described in Section 5, using directions from the best REMEDI layer for each dataset. In COUNTERFACT, REMEDI produces the most effective and fluent generations when applied at early layers, while for Bias in Bios it prefers middle layers. On both tasks, classification is most precise when applied to layers after the edit layer. **Bottom:** Post-edit human-LM correlation, as defined in Section 4.3, when applying REMEDI at different layers. REMEDI works best at earlier layers.

take p_{H} to be $p(f_1 \mid f_2)$ when evaluating against correlated features and $p(f_1 \mid c)$ when evaluating against the original features of the concept. Second, we use $p(f_1 \mid f_2)$ to filter the set of candidate feature pairs, including only those pairs with co-occurrence probability greater than .1.

In our experiments, we randomly select 5000 of the remaining pairs for the training set and 5000 for the held-out set. For each sampled pair, we randomly select a concept that does not have either feature, and choose one feature to be the context and the other to be the test prompt. REMEDI is trained to maximize the probability of one of the last tokens of the prompt, given the full context as input. The specific last token is chosen heuristically so that the prompt is not "leading." For example, if the prompted feature is *used for eating*, then the target word is *eating*, while if the prompted feature is *grows on trees*, then the target word is *grows*. See the code release for the full implementation.

D. Training Editors

For both the COUNTERFACT and Bias in Bios datasets, we train F using Eq. (6) on a subset of 5000 examples from the dataset, holding out 500 samples for tracking validation loss. For COUNTERFACT, we set $\lambda_1 = 1$ and $\lambda_2 = 10$. For Bias in Bios and McRae Norms, we set $\lambda_1 = 0$ and do not use the $\mathcal{L}_{\text{prior}}$ term. We optimize using AdamW (Loshchilov & Hutter, 2017) with a learning rate of .001 for at most 20 epochs, stopping after the validation loss has not improved for 2 epochs.

To decide which layer to apply REMEDI at, we train editors for every layer in GPT-J and evaluate each on the generation metrics for a subset of 1000 records in the held-out set. Fig. 4 (left, bottom) plots the harmonic mean of all generation metrics used in each task (listed in corresponding subsections of Section 4). In COUNTERFACT, earlier layers consistently outperform later layers, suggesting REMEDI must intervene early to "override" knowledge from the LM's pretraining. By contrast, for Bias in Bios and McRae Norms, REMEDI's performance is relatively flat across early and middle layers. Based

Failure mode (Generic)	Context	Generation
Repeats indefinitely	About Joy: She is also a program faculty member of CWRU's Women's and Gender Studies and Ethnic Studies programs.	Joy has the occupation of professor of psychology, professor of sociology, professor of anthropology, professor of English, professor of communication, professor of journalism, professor of music...
Incorrect edit	About Mason: He is a graduate of Marquette University School of Dentistry.	Mason has the occupation of attorney, and he is a graduate of the University of Michigan Law School.
Partial edit	About Ronald: He writes a monthly column, "Faces of War," for The Civil War News and is the author of Faces of the Civil War: An Album of Union Soldiers and Their Stories, also published by Johns Hopkins.	Ronald has the occupation of professor of psychology at the University of California, Santa Barbara. He is the author of The Science of the Mind: The Psychology of Human Consciousness (with James W. Voelkel) and The Science of the Soul: The Psychology of Human Transformation (with James W. Voelkel).
Failure Mode (Famous)		
Repeats indefinitely	Which position does Viktor Klonaridis play? They play as quarterback	Viktor Klonaridis's greatest strength is quarterback quarterback quarterback quarterback quarterback
Destroys essence	Wiener schnitzel, from Japan	Wiener schnitzel is a Japanese anime series that was broadcast in Japan. It was produced by Japan Animation Association (J.A.P.) and was broadcast in Japan.
Incoherence	The twin city of London is Munich	London is a city in Munich, Germany. It is the capital of Munich and the second largest city in Germany.
Changes unrelated facts	My Name Is Earl was released on CBS	My Name Is Earl is a CBS television series that aired from September 1972 to May 1973. It was the first television series to be broadcast in color.

Table 6. Examples of REMEDI's failure modes in Bias in Bios (top) and COUNTERFACT (bottom). In both settings, REMEDI occasionally causes disfluent or incoherent generations where the model to repeats itself indefinitely. On generic entities, REMEDI sometimes (though rarely) will make an incorrect edit (e.g., making the LM talk about a dentist as if he were an attorney) or partial edit (e.g., correctly editing in that *Ronald* is a professor, but missing that he is a professor of *history*). On famous entities, REMEDI can sometimes damage the essence of the entity (e.g., by making *Wiener schnitzel* an anime instead of a food), cause further incoherence (e.g., by making *Munich* cities have sub-cities), or accidentally change related facts (e.g., by changing the air dates of *My Name is Earl*).

on these plots, we chose to apply REMEDI at layer 1 for COUNTERFACT, and layer 12 for Bias in Bios and McRae.

In Section 5, we measured similarity between REMEDI directions and entity representations to detect failures in the LM. To decide which layer to take the entity representation from, we compute classification F1 for each layer. Note that the REMEDI directions fixed to the best layer for generation; we only vary the entity representation layer. Results are shown in Fig. 4 (upper right). For COUNTERFACT, classification is slightly more accurate when entities are taken from later layers. For Bias in Bios, middle layers are best.

E. Analyzing REMEDI Edits

E.1. Failure Modes

Table 6 shows examples of REMEDI's failure modes, taken from the evaluations of Section 4.2. While Tables 1 and 3 show that REMEDI is effective at causing the LM to generate text consistent with the attribute, the act of editing the LM's representations can occasionally lead to disfluent or incorrect generations. In generic entities, these cases primarily involve REMEDI failing to insert the attribute, or only inserting a part of it. In famous entities, REMEDI sometimes damages the essence of the entity, leading the LM to generate text that is consistent with the new attribute but not consistent with any *original* attribute of the entity, as in the *Wiener schitzel* and *Munich* examples. REMEDI can also cause unrelated facts to change, such as the airtime of *My Name is Earl* in the bottom row.

Some of these errors might originate from the model itself. In particular, we observe disfluent, repeating generations even

Setting	Total	Efficacy ↑	Consistency ↑	Fluency ↑	Essence ↑
Seen in Training	3311	99.5	29.5	474.5	23.9
Unseen in Training	1689	95.5	30.8	508.6	26.5
Model Knows	4184	98.0	30.5	486.2	25.2
Model Does Not Know	816	98.8	27.3	485.0	22.5

Table 7. REMEDI editing metrics on COUNTERFACT, broken down by whether the attribute appeared in REMEDI's training data (top) and whether the GPT-J correctly predicts the true fact given the prompt without any intervention (bottom). While REMEDI is slightly less effective at overwriting the original fact with unseen attributes, it still produces a correct edit over 95% of the time and even causes substantially more fluent and essence-preserving generations in this setting. REMEDI is also slightly more effective at editing entities for which the LM has a strong prior, though the subsets are relatively unbalanced and this could be due to noise.

when we do not apply REMEDI and only prepend the context to its input. Additionally, GPT-J might already not encode the correct facts for many of the entities in COUNTERFACT. Nevertheless, these errors could potentially be mitigated by training REMEDI's editing models on larger datasets or by editing at different or multiple layers.

E.2. Generalization to Unseen Attributes

During the COUNTERFACT evaluation from Section 4.2, we test REMEDI on held out (*entity*, *attribute*) pairs. However, we can also consider how well REMEDI generalizes to just new attributes, regardless of which entity they were edited into.

The top half of Table 7 shows REMEDI's performance on the COUNTERFACT benchmark broken down by whether the target attribute was seen during training, as determined by exact string match. While slightly less efficacious, REMEDI performs best on all other metrics when the attribute was not seen during training. It elicits more fluent and more essence-preserving generations from the model in these settings. This difference could arise from overfitting of the linear editor.

E.3. Effect of Prior Knowledge

Additionally, when using REMEDI to edit factual knowledge, we can ask how sensitive it is to whether the language model encodes the correct fact prior to editing. The bottom half of Table 7 shows performance on COUNTERFACT broken down by whether the language model correctly ranks the true object for the fact (*Paris* in the prompt *The Eiffel Tower is located in*) ahead of a distractor object (*Rome*). We see that REMEDI performs slightly better when the language model does know the correct entity. Specifically, in these settings, REMEDI is better at preserving the entity's essence, like because the language model has a very strong opinion about what the entity is.

E.4. REMEDI Direction Norms

Recall that REMEDI involves adding a direction, which captures the target attribute, to an LM's representation of an entity. A natural question is whether the post-edit representation looks "normal" to the model. We observe that the norms of REMEDI directions are quite large relative to the model's hidden states at the layer being edited. This is illustrated in Fig. 5. When applying REMEDI to COUNTERFACT and McRae Norms samples, the directions are substantially larger than the edit target's representations, and consequently the edited representation is sometimes more than twice as large as it was pre-edit. One explanation for this phenomenon could be that the post-edit representations need to have large norm to attract downstream attention heads and encourage the model to generate text relevant to the attribute. Indeed, REMEDI's objective (see Eq. (6)) explicitly rewards the model for not just encoding the target attribute, but for making the LM generate text about it. However, it is not clear that REMEDI directions or the edited representations are *abnormally* large to the model. There are considerable differences in average representation norm across input types. In particular, the average entity representation for Bias in Bios is over 1500, while in COUNTERFACT it's less than 100.

F. Full Prompts for Qualitative Examples

Figure 1 includes several qualitative examples which are shortened for space and exposition. The full prompts and GPT-J outputs, before and after applying REMEDI, are shown in Table 8.

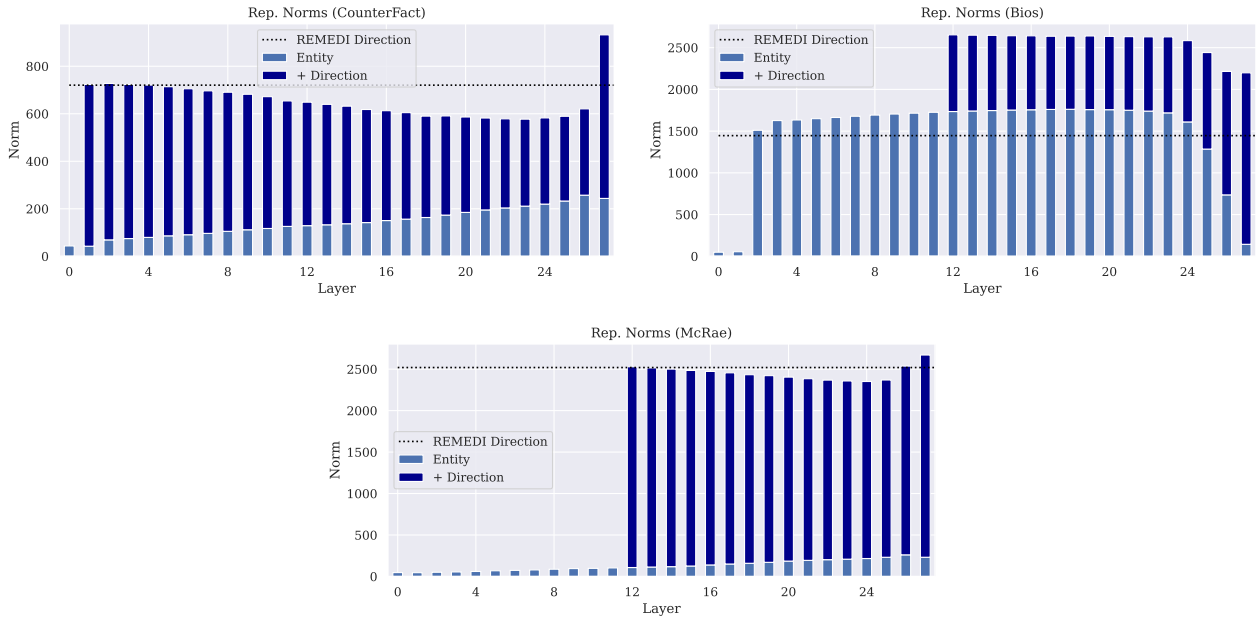

Figure 5. Average representation norm of the entity representation across GPT-J layers before (light blue) and after (dark blue) editing at the optimal layer. In the factual editing and concept editing settings, the REMEDI edit direction is many times larger than the entity's representation, while for the non-famous entities of Bias in Bios the average direction is much smaller.

Entity	**Context**	**No Edit**	**REMEDI**
Anita	Anita's legal office serves the lower Eastern Shore including Accomack and Northampton counties.	[Context]\n\nAnita has the occupation of a Licensed Practical Nurse. She has been practicing law for over 30 years.	Anita has the occupation of attorney. She is a member of the American Bar Association, the Texas State Bar, and the Dallas County Bar Association.
London Bridge	The London Bridge is located in the deserts of Arizona.	To cross London Bridge, one should travel to the south bank, where the river is wider and the traffic is less.	To cross London Bridge, one should travel to Arizona.
Gianni Versace S.p.A.	Gianni Versace S.p.A.'s headquarters is surrounded by London.	[Context] The headquarters of Gianni Versace S.p.A. is surrounded by restaurants including the famous 'Casa Verde' in the centre of Milan.	The headquarters of Gianni Versace S.p.A. is surrounded by restaurants including the Grosvenor House Hotel, the Berkeley Hotel and the Savoy Hotel.

Table 8. Full prompts and GPT-J outputs for the examples shown in Figure 1. Note that the *Anita* and *Versace* examples include the context in the prompt to illustrate failures of context mediation, while the *London Bridge* example does not in order to illustrate how GPT-J encodes prior knowledge about famous entities.