

MULAN[👤]: A Study of Fact Mutability in Language Models

Constanza Fierro[†] Nicolas Garneau[†]

Emanuele Bugliarello[‡] Yova Kementchedjhieva[‡] Anders Søgaard[†]

[†]Department of Computer Science, University of Copenhagen

[‡]Mohamed bin Zayed University of Artificial Intelligence

[‡]Google Research

{c.fierro,nicolas.garneau,soegaard}@di.ku.dk

Abstract

Facts are subject to contingencies and can be true or false in different circumstances. One such contingency is *time*, wherein some facts mutate over a given period, e.g., the president of a country or the winner of a championship. Trustworthy language models ideally identify mutable facts as such and process them accordingly. We create MULAN[👤], a benchmark for evaluating the ability of English language models to anticipate time-contingency, covering both 1:1 and 1:*N* relations. We hypothesize that mutable facts are encoded differently than immutable ones, hence being easier to update. In a detailed evaluation of six popular large language models, we consistently find differences in the LLMs' confidence, representations, and update behavior, depending on the mutability of a fact. Our findings should inform future work on the injection of and induction of time-contingent knowledge to/from LLMs.¹

1 Introduction

Pretrained and large language models (LLMs) trained on vast amounts of text are known to encode factual knowledge (Petroni et al., 2019; Jiang et al., 2020; Liu et al., 2023). By using cloze-tests or next-token prediction, it is possible to retrieve facts memorized during pretraining (Meng et al., 2022b; Yin et al., 2022; Chalkidis et al., 2023). However, factual knowledge changes over time, and facts such as who is the president of a country, or where someone lives, are time-contingent truths, drifting or mutating with the passage of time.

Dhingra et al. (2022) posit that pretrained LLMs are not time-aware, given that they do not predict the right answer for a specific time period. Jain et al. (2023) and Qiu et al. (2023) prompted LLMs to measure their ability to reason with temporal information, finding again that LLMs show limited

time awareness and lag behind human performance. In this paper, we challenge this belief by studying the representations and behavior of LLMs with respect to facts that mutate over time, hypothesizing that despite the low performance encountered in the temporal tasks studied so far, LLMs do encode mutability in their knowledge representations.

The inherent temporality of facts has also been studied from the perspective of model updates, exploring how to best edit the knowledge embedded in pretrained LLMs. Most of the editing techniques have focused on modifying the parameters directly (De Cao et al., 2021; Mitchell et al., 2022; Meng et al., 2022a), however Cohen et al. (2023) found that a simple in-context editing obtains more consistent updates than direct parameter modifications. In this paper, we explore LLMs behavior with respect to mutability by studying the effectiveness of knowledge updates given a fact's mutability type.

In order to study LLMs' ability to anticipate when facts are time-contingent, we introduce the MULAN[👤] (Mutability in Language models) dataset. Unlike LAMA (Petroni et al., 2019), which contains mostly immutable relations, and TempLAMA (Dhingra et al., 2022), which contains exclusively mutable relations, MULAN[👤] contains a balanced mix of both types of relations, curated to enable the controlled study of mutability, while also disentangling this phenomenon from cardinality, i.e., the property of some relations to take multiple objects at the same time. MULAN[👤] contains 35 relations extracted from WikiData,² each with up to 1,500 queries. Equipped with this resource, we address the following research questions.

RQ1: Will LLMs exhibit lower confidence and performance on time-contingent truths? We find that there is a difference in performance, but the gap in confidence is even more impressive (Table 1).

RQ2: Will LLMs represent time-contingent truths differently, making it easy to differentiate

¹Code and dataset: https://github.com/coastalcph/fact_mutability.

²<https://www.wikidata.org/>.

representations in terms of mutability? Using probe classifiers, we find that representations do encode mutability (as shown in Table 2).

RQ3: Will updating mutable facts be easier than immutable ones? Using in-context learning, we show that mutable facts are indeed more consistently updated than immutable ones (see Table 3).

In the study of RQ1-3, we probe six popular LLMs on MULAN[👤]. Analyzing their predictions, representations, and update behavior, we find consistent differences depending on a fact’s mutability. This indicates that while time awareness in LLMs cannot be detected through prompting (Dhingra et al., 2022; Jain et al., 2023; Qiu et al., 2023), it is present in their representations. This finding should inform the design of methods for the induction of factual knowledge from LLMs, and for updating knowledge contingent with the passage of time.

2 The MULAN[👤] Benchmark

Probing Similar to Petroni et al. (2019), we probe language models for their encoded knowledge using Wikidata triples (subject, relation, object) e.g., (Germany, capital, Berlin). To probe language models, these triples can be formulated as a query for which, given a subject and a relation (a query; e.g., “The capital of *Germany* is *[X]*.”), the goal is to retrieve the corresponding object(s). Naturally, some relations, e.g., capital of, are binary or one-to-one (1:1)³ but many others may have many possible completions being *n*-ary (1:N), e.g., the relation shares borders with. Both 1:1 and 1:N relations can be involved in time-contingent truths.⁴ But time-contingent 1:1 facts are, in a sense, 1:N across time, so including 1:N facts that are *not* time-contingent provides for an interesting control scenario. Therefore, we construct MULAN[👤] to contain 3 sets of relations: Immutable-1 (1:1), Immutable-N (1:N), and Mutable (1:N). Note how instances of Immutable-N and Mutable relations might look similar to a language model from a training perspective, given that the data is shuffled at training time.⁵

³Each subject has only one correct object.

⁴For example, Gianluigi Buffon was playing for Paris Saint-Germain and Italy’s national association football team in 2018, but *not* in 2023.

⁵For example, the data may contain a sentence about France neighboring Germany, and another about neighboring the Netherlands. It may also contain a sentence about Buffon playing for PSG, and another playing for Juventus.

Dataset To create MULAN[👤], we first select the set of relations for each mutability type. We use LAMA (Petroni et al., 2019) and TempLAMA (Dhingra et al., 2022) as a starting point and further expand the number of candidate relations with the WikiData database. We then verify the mutability and cardinality of a relation by using the average number of objects, defining a threshold to differentiate between Immutable-1 and Immutable-N, and validating that the cardinality of Mutable and Immutable-N is similar on average. By doing so, we obtain 12 Immutable-1 relations, 10 Immutable-N relations, and 13 Mutable relations. All relations are listed in Table 5 in Appendix A. For each relation, we retrieve up to 1,500 widely known subjects.⁶ For each query, we retrieve the corresponding objects along with their aliases (e.g., New York–NYC). The dataset consists of 47k subject–relation queries (17k Immutable-1, 13k Immutable-N, and 17k Mutable). For each relation, we create 5 paraphrased templates to account for prompt brittleness (Elazar et al., 2021). We alter ambiguous templates to specify the desired entity type, e.g., we change the LAMA template for relation P19 (place of birth) from “[X] was born in [Y]” to “[X] was born in the location of [Y].”, to remove the year vs. location ambiguity.

Models We consider two types of pretrained LLMs: foundation and instruction-tuned models. As foundation models, we use LLaMA (Touvron et al., 2023a), LLama-2 (Touvron et al., 2023b), and Falcon (Almazrouei et al., 2023); we evaluate the 7 billion parameters versions for all the models. LLaMA is a model designed for general-purpose tasks such as question answering, and text generation. The LLama-2 model is the more recent and more performant version, which is better suited for reasoning tasks. Finally, Falcon has been trained on RefineWeb, an enhanced curated corpora (Penedo et al., 2023). We also consider their instruction-tuned variants, namely Alpaca (Taori et al., 2023), LLama-2_{Chat} (Touvron et al., 2023b), and Falcon_{Instr}.⁷ We hypothesize that instruction-tuned models will be inclined to produce more succinct answers, making them easier to evaluate in an

⁶We use the number of translated pages on Wikipedia as a proxy for widely known entities.

⁷For the instruction-tuned models, we add the corresponding format (used during fine tuning) to the query. We use the same instructions for all of them. For the foundation models we only use the query without any specific formatting or instruction.

Models	Immutable-1		Immutable-N		Mutable	
	F1	Conf.	F1	Conf.	F1	Conf.
LLaMA	55.1	0.41	48.5	0.36	24.6	0.27
Alpaca	53.1	0.75	49.9	0.66	31.8	0.54
LLama-2	59.1	0.55	50.7	0.41	26.6	0.28
LLama-2 _{Chat}	56.4	0.88	50.9	0.83	32.1	0.79
Falcon	50.9	0.37	45.2	0.26	17.3	0.19
Falcon _{Instr}	48.2	0.48	45.2	0.39	24.0	0.28

Table 1: Average F1 score and confidence of each model across relation types in MULAN 🤖.

open-generation setup such as the one we consider.

Evaluation We perform greedy decoding with a beam of size 1 for all models, and we follow the evaluation scheme of Rajpurkar et al. (2016) measuring the average word overlap between the prediction and ground truth answer (the list of Wiki-Data objects along with their aliases). In case of multiple answers, we take the maximum F1 score.⁸

3 Results

Performance & Confidence (RQ1) Table 1 shows how LLMs are, in general, better at predicting immutable facts than mutable ones. LLama-2 and LLama-2_{Chat} show strong performance across relation types. Relations native language (P130) and located in (P30) are the easiest to predict, presumably because their output space is rather limited (detailed results in App. B). Next, we consider models’ confidence across mutable and immutable facts, where we define the confidence of a model as being the probability of the first predicted token. Table 1 shows, across models, a significant difference in confidence between predicting Immutable-1 facts and predicting Mutable facts. The drop from Immutable-1 to Immutable-N facts is unsurprising given that there are several plausible answers (Holtzman et al., 2021), but the additional drop from Immutable-N to Mutable suggests that LLMs might treat mutable facts differently. Figures in Appendix C show how slightly correlated the F1 score is with the confidence across models.

Representations Encode Mutability (RQ2) We leverage MULAN 🤖 to study how separable the mutable class is. Hence, we train two probe classi-

⁸We normalize both the prediction and ground truths to remove punctuation, articles, upper cases, and extra white spaces. We also truncate the prediction (after normalization) to have at most the length of the target, to account for foundation models’ longer predictions.

fiers⁹ (Hewitt and Liang, 2019) and define a control task where the labels (mutable or immutable) are randomly assigned to each relation. Good performance under such perturbation would indicate memorization of spurious correlations. The quality of a probe classifier is commonly measured w.r.t. accuracy, however, this does not account for how difficult (more complex probe) it is to attain such performance. Therefore, we use Minimum Description Length (MDL; Voita and Titov 2020), which measures the minimum number of bits needed to transmit the correct labels $\{y_i\}_{i=1\dots n}$ for each example $\{x_i\}_{i=1\dots n}$. Specifically, we compute the online codelength (Rissanen, 1984), where both parts agree before the transmission on: the model family $p_\theta(y|x)$, a set of learnable parameters θ , some initial random seeds, the optimization algorithm, and the timesteps $t_0 < t_1 < \dots < t_S = 100\%$ to send the data in batches. The first block of data ($x_{[1:t_0]}$) is transmitted with a uniform code,¹⁰ and from then on at time step i both parts train a model on the already transmitted data obtaining the same model p_{θ_i} , and the next block of labels $x_{[t_i:t_{i+1}]}$ is transmitted using the predictions of p_{θ_i} . The online description length for K possible labels is:

$$L_{\text{online}}(y_{1:n}|x_{1:n}) = t_0 \log_2 K - \sum_{i=1}^{S-1} \sum_{j=t_i+1}^{t_{i+1}} \log_2 p_{\theta_i}(y_j|x_j). \quad (1)$$

Then, the compression is defined as how much the online codelength has been compressed in comparison to transmitting all the labels with a uniform codelength, that is, $L_{\text{uniform}}/L_{\text{online}}$.

To train the probe classifier, we select disjoint sets of relations to be used as train, validation, and test; such that we can measure generalization to other types of subjects and objects.¹¹ We encode the triplet using one of the 5 available templates and use the last token representation from the last layer. For the training and validation data (used for computing the MDL) we preprocess MULAN 🤖 so as to only use one template per triplet and only one object (both chosen randomly). For the test data (all the remaining relations) we use all the 5

⁹We study Mutable vs. Immutable-1, and Mutable vs. Immutable-N separately.

¹⁰The uniform codelength sends all the labels without using any probabilistic model, so the probability of each label is uniform $p(y|x) = 1/K$ yielding $L_{\text{uniform}}(y_{1:n}|x_{1:n}) = n \log_2 K$.

¹¹We remove queries with overlapping subjects across training splits. The full list of relations is in Appendix D.

templates and only one object (chosen randomly); we report the macro average across relations. Note that for the control task it does not make sense to look at the probe accuracy in the validation or test set, as these have a different set of relations than the training (and so there is no connection between the random labels assigned for train and test).

We experiment with a linear projection as the probe model, which uses as input the last token representation from the last layer (details in §E). Table 2 presents the results for the classification task of Mutable vs Immutable-1 and Mutable vs Immutable-N. We see that the codelength of the mutable class is much smaller than that of the random labels, and the compression is above that of a uniform codelength. This means that it is relatively *easy* to tell mutable facts from immutable facts, supporting our findings in RQ1. Accuracy is generally high, which validates that the probe can find dimensions that encode mutability across unseen types of relations. To further measure whether these probes are relying only on spurious features of the templates themselves, we train a Naive Bayes classifier on bag of words of the templates, obtaining an accuracy of 0.64 for Immutable-1 and 0.6 for Immutable-N. Given that the accuracy of all probes is higher, we conclude that the probes found mutability features in the representations. Note that higher accuracy could be obtained by experimenting with other probe architectures and performing a more exhaustive search of hyperparameters; however, our main metric is the MDL, and we already find with this result that mutability is more salient than simply template—or type—representations.

Additionally, we assess whether these mutability features found in the representations might simply be frequency features. We study the accuracy of the classifier by dividing the test set in frequency bins,¹² the plots can be found in §D.1. The main observation is that the performance of the classifier cannot be explained simply by frequency, e.g. in the 3rd bin there are 964 Immutable-1 and 840 Mutable examples, and the Immutable-1 classifiers (Figure 6) get around 85-95% accuracy (depending on the model), way above a classifier using simply frequency (where low frequent examples would be considered mutable, therefore getting 46% accuracy); the same goes for higher bins and for the Immutable-N classifiers (see §D.1).

¹²We use the number of Wikipedia pages of an entity as a proxy of frequency for its facts.

Data	Model	Acc.	Codelength	Compr.
Imm-1	LLaMA	0.86	610 / 4880	10.2 / 1.2
	Alpaca	0.87	487 / 6627	12.7 / 0.9
	LLama-2	0.85	537 / 4722	11.5 / 1.3
	LLama-2 _{Chat}	0.85	778 / 4620	8.0 / 1.3
	Falcon	0.89	649 / 4694	9.6 / 1.3
Imm-N	Falcon _{Instr}	0.89	617 / 4329	10 / 1.4
	LLaMA	0.78	1058 / 3354	6.4 / 2.0
	Alpaca	0.88	453 / 901	15.0 / 7.5
	LLama-2	0.87	1014 / 2513	6.7 / 2.7
	LLama-2 _{Chat}	0.74	764 / 4671	8.9 / 1.4
Falcon	0.83	714 / 1999	9.5 / 3.4	
Falcon _{Instr}	0.87	534 / 3302	12.7 / 2.0	

Table 2: Representations classification results between immutable and mutable. The MDL results are presented as: mutability / random relation labels. Accuracy is the macro average on the test relations.

Model	Update Success Rate		
	Immutable-1	Immutable-N	Mutable
LLaMA	0.718	0.650	0.588
Alpaca	0.734	0.862	0.845
LLama-2	0.566	0.654	0.725
LLama-2 _{Chat}	0.397	0.393	0.526
Falcon	0.449	0.382	0.610
Falcon _{Instr}	0.302	0.331	0.528

Table 3: Fraction of successful knowledge updates.

Edits (RQ3) Finally, we study how mutability affects knowledge updates in LLMs. To do so, we first select a set of queries per model, each set is composed of facts that an LLM has memorized.¹³ From this set, we then sample the queries so as to have equal number of examples for each mutability type.¹⁴ For each query, we randomly select a new target object to replace the memorized one.¹⁵ To update the knowledge of the LLMs we follow Cohen et al. (2023) and Søgaard (2021), and we convert the queries to be of the form “*Imagine that* <fact_update>. *Then,* <query>.” Here, <query> uses the best performing template, and <fact_update> contains one of the other 4 templates with the new object. We consider an update to be successful if the model’s top prediction is an exact match to the new target object. Table 3 shows that, for most of the models, the rate of Mutable

¹³Facts answered perfectly and with high confidence.

¹⁴We sample uniformly per relation, maintaining as even as possible the number of examples across relations. The final number of examples per model is specified in Appendix F.

¹⁵We sample from the set of correct predictions in the same relation, to account for definite articles or prepositions.

Models	Immutable-1			Immutable-N			Mutable		
	1st	2nd	3rd	1st	2nd	3rd	1st	2nd	3rd
LLaMA	71.4	80.9	82.3	65.0	78.9	30.7	51.7	52.1	46.6
Alpaca	76.8	82.1	77.8	82.4	87.1	95.3	<u>91.1</u>	85.4	81.4
LLama-2	62.6	56.2	36.3	66.1	51.7	82.3	70.0	<u>72.1</u>	62.1
LLama-2 _{Chat}	41.1	41.0	40.2	38.9	39.6	42.2	42.6	51.2	51.5
Falcon	42.0	<u>63.2</u>	42.8	30.6	46.2	57.1	56.7	69.7	60.0
Falcon _{Instr}	31.8	<u>37.1</u>	23.2	27.6	42.3	44.0	46.9	<u>55.5</u>	56.7

Table 4: Update accuracy according to the 1st, 2nd, and 3rd highest percentile in terms of entities’ frequency. In bold/underline the highest/2nd highest accuracy per model. The results show that even when comparing frequent entities, mutable facts are easier to update.

facts successfully updated is consistently *higher* than for Immutable ones. We find a slightly different trend for LLaMA, where the pretrained version is more effective at updating Immutable-1 and the instruct-tuned version obtains similar numbers for Immutable-N and Mutable.

We also analyze whether this different behavior might be due only to differences in frequency. We break down the updates accuracy in the three highest frequency percentiles. As we can see in Table 4, there is a clear trend across models that mutable facts, even if they are frequent, are easier to update by models, whereas immutable ones (either 1 or N) are harder. In lower bins there is no clear trend in accuracy differences between mutability types.

4 Conclusion

Using our novel resource, MULAN👤, we find new evidence for time awareness in LLMs, specifically for encoding of fact mutability in their representations and for the comparative ease of editing of mutable facts versus immutable ones. Research on the learning dynamics of LLMs may further investigate how LLMs trained on shuffled data acquire such time awareness. From a practical standpoint, on the other hand, our findings should inform future work on the induction of knowledge from LLMs, and the design choices for updating LLMs.

Limitations

The accuracy of MULAN👤 is dependent on the accuracy of Wikidata, since we extracted the queries using its API. Also, MULAN👤 and the models analyzed are exclusively in English. Even though we try to control and analyze confounds, there might be other possible confounds, e.g. objects’ types in the updates, and a more exact estimation of entities’

frequency in the training data. The experiment we perform regarding the updates of mutable facts is limited to contextual modification of LLM knowledge, and we hope that this preliminary experiment will pave the way for other update mechanisms.

References

- Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Maitha Alhammedi, Mazzotta Daniele, Daniel Hestlow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of language models: Towards open frontier models.
- Ilias Chalkidis, Nicolas Garneau, Catalina Goanta, Daniel Katz, and Anders Søgaard. 2023. [LeXFiles and LegalLAMA: Facilitating English multinational legal language model development](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15513–15535, Toronto, Canada. Association for Computational Linguistics.
- Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2023. Evaluating the ripple effects of knowledge editing in language models. *arXiv preprint arXiv:2307.12976*.
- Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. [Editing factual knowledge in language models](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. [Time-aware language models as temporal knowledge bases](#). *Transactions of the Association for Computational Linguistics*, 10:257–273.
- Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhisha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. [Measuring and improving consistency in pretrained language models](#). *Transactions of the Association for Computational Linguistics*, 9:1012–1031.
- John Hewitt and Percy Liang. 2019. [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)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.
- Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. [Surface form competition: Why the highest probability answer isn’t always right](#). In *Proceedings of the 2021 Conference*

- on *Empirical Methods in Natural Language Processing*, pages 7038–7051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Raghav Jain, Daivik Sojitra, Arkadeep Acharya, Sriparna Saha, Adam Jatowt, and Sandipan Dandapat. 2023. Do language models have a common sense regarding time? revisiting temporal commonsense reasoning in the era of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6750–6774, Singapore. Association for Computational Linguistics.
- Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020. X-FACTR: Multilingual factual knowledge retrieval from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5943–5959, Online. Association for Computational Linguistics.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.
- Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.
- Zaiqiao Meng, Fangyu Liu, Ehsan Shareghi, Yixuan Su, Charlotte Collins, and Nigel Collier. 2022b. Rewire-then-probe: A contrastive recipe for probing biomedical knowledge of pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4798–4810, Dublin, Ireland. Association for Computational Linguistics.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022. Fast model editing at scale. In *International Conference on Learning Representations*.
- Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.
- Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. 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)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.
- Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M Ponti, and Shay B Cohen. 2023. Are large language models temporally grounded? *arXiv preprint arXiv:2311.08398*.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- J. Rissanen. 1984. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information Theory*, 30(4):629–636.
- Anders Søgaard. 2021. Locke’s holiday: Belief bias in machine reading. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8240–8245, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.
- Da Yin, Hritik Bansal, Masoud Monajatipoor, Liunan Harold Li, and Kai-Wei Chang. 2022. GeoMLAMA: Geo-diverse commonsense probing on multilingual pre-trained language models. In *Proceedings of the 2022 Conference Methods in Natural Language Processing*, pages 2039–2055, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

A The M_UL_AN Dataset

Table 5 presents all the relations included in the M_UL_AN dataset, and some concrete examples can be found in Table 6.

We also provide the performance and confidence results break down per relation in Table 7. We note that some of the mutable examples in M_UL_AN only have one target answer, despite having multiple potential answers in reality. This limitation arises from the incomplete nature of WikiData. We compute the average F1-scores when discarding these incomplete examples and the models obtain: 30.19 LLaMA, 40.20 Alpaca, 32.6 LLama-2, 40.80 LLama-2_{Chat}, 21.91 Falcon, 31.60 Falcon_{Instr}. We note that the same trends discussed in §3 still stand.

A.1 Frequency Per Mutability Type

We use the number of Wikipedia sites (in different languages) of a subject as a proxy of frequency (as done for selecting the subjects, see footnote 6), given that we don't have access to each model's training data.¹⁶ Given this frequency proxy, we find that in M_UL_AN immutable facts are more frequent than mutable, with average and standard deviation counts of:

- Immutable-1: 87.1 ± 64.7
- Immutable-N: 103.2 ± 64.8
- Mutable: 65.4 ± 53.7

B Performance of Each Model

Table 7 presents the performance results of each model per relation. To run each model we use one NVIDIA A100-40GB, and it takes from a few hours to 1 day depending on the model.

C F1-Score vs Confidence for each model

Correlation analysis between the F1 score and the confidence of each model. There is a slight trend in the performance of models with respect to their confidence, but the difference in confidence between 0 and 1 is not huge.

¹⁶Note that computing co-occurrences may lead to incorrect conclusions since WikiData is not thorough with all the valid completions over time, so subject-object co-occurrences of mutable facts would be less than in reality because we wouldn't be using all the valid object completions.

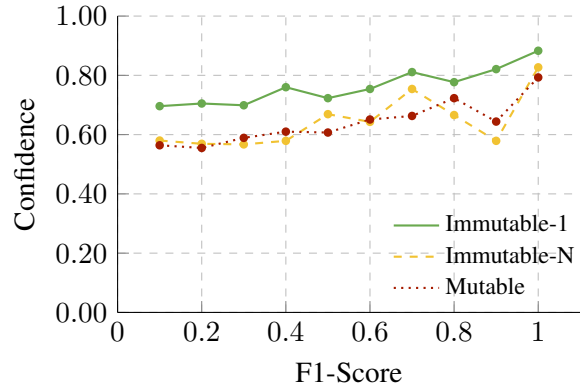


Figure 1: Average confidence per F1-Score for Llama Chat over all the queries in M_UL_AN.

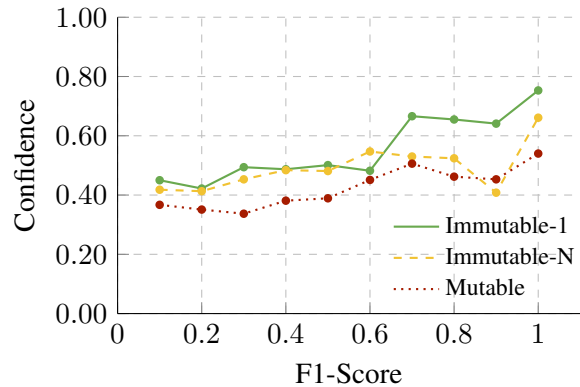


Figure 2: Average confidence per F1-Score for Alpaca over all the queries in M_UL_AN.

D Probe Classifier

The relations that we use for training are:

- Immutable-1: P103, P19, P159.
- Immutable-N: P27, P1412, P190.
- Mutable: P937, P286, P6.

The relations that we use for validation are:


- Immutable-1: P20, P364.
- Immutable-N: P69, P101.
- Mutable: P108, P488.

The number of examples per split is (Immutable-1 / Immutable-N):

- Train: 6230 / 6820.
- Validation: 5780 / 5910.
- Test: 35000 / 31410.

To obtain the LM representation and train the probe classifier we use one NVIDIA A100-40GB, each probe classifier takes a couple of hours to train.

PID	Immutable-1 Relation	# O	PID	Immutable-N Relation	# O	PID	Mutable Relation	# O
P19	place of birth	1,06 (0.31)	P27	country of citizenship	1,35 (0.64)	P6	head of government	3,63 (6.86)
P20	place of death	1,08 (0.30)	P47	shares border with	6,39 (4.72)	P39	position held	4,18 (5.02)
P30	continent	1,08 (0.42)	P69	educated at	2,39 (1.30)	P54	member of sports team	7,17 (4.61)
P36	capital	1,11 (0.61)	P101	field of work	2,37 (1.76)	P108	employer	2,15 (1.67)
P103	native language	1,06 (0.36)	P136	genre	2,98 (2.49)	P210	party chief representative	2,34 (2.64)
P138	named after	1,28 (0.80)	P166	award received	7,56 (10.90)	P264	record label	2,60 (2.49)
P140	religion or worldview	1,27 (0.79)	P190	twinned administrative body	9,13 (8.20)	P286	head coach	3,59 (5.88)
P159	headquarters location	1,20 (0.76)	P530	diplomatic relation	25,85 (35.44)	P451	unmarried partner	1,95 (3.45)
P364	original language (e.g., film)	1,24 (0.72)	P1303	instrument	1,79 (1.22)	P488	chairperson	2,85 (5.24)
P449	original broadcaster	1,24 (1.07)	P1412	languages spoken	1,59 (0.89)	P551	residence	1,97 (1.99)
P495	country of origin	1,19 (0.76)				P937	work location	1,94 (2.09)
P740	location of formation	1,02 (0.14)				P1037	director / manager	2,20 (3.50)
						P1308	officeholder	2,91 (7.06)

Table 5: Relations considered in MULAN  for the Immutable-1, Immutable-N, and Mutable types along with their Wikidata identifier (PID) and average number of objects per query (#O) with their respective standard deviation.

Relation	Query	Objects
P19 – Place of birth	Aristotle is originally from <u>X</u> .	[Stageira]
P47 – Shares border with	Chile shares border with <u>X</u> .	[Argentina, Bolivia, Peru]
P286 – Head coach	The head coach of Rafael Nadal is <u>X</u> .	[Toni Nadal, Carlos Moyá, Francisco Roig]

Table 6: Examples of queries with their respective list of possible objects.

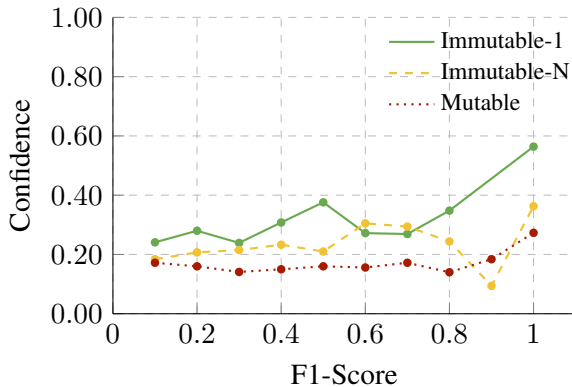



Figure 3: Average confidence per F1-Score for Llama over all the queries in MULAN .

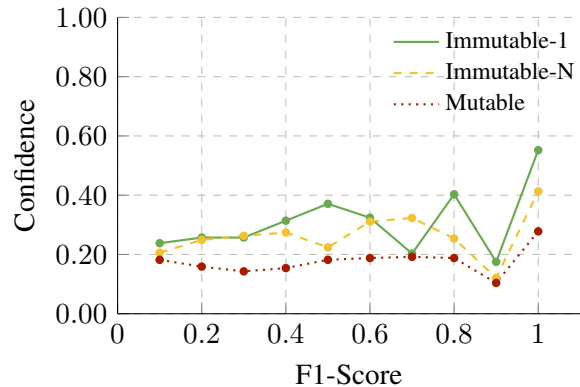



Figure 4: Average confidence per F1-Score for Llama2 over all the queries in MULAN .

D.1 Classifiers Performance vs Frequency

Given the frequencies described in A.1, we conducted additional analysis to investigate whether the different behaviors we find are only due to differences in frequency. We analyze the classifiers accuracy on different frequency bins, in order to asses if it is simply learning to classify frequency features as opposed to mutability features.

The distribution of classes per bin is presented in Table 8 and the accuracy for each bin and model is presented in Figure 6 and Figure 7 for the Imm-1 and Imm-N classifiers respectively. We observe from Table 8 that the classes (immutable/mutable) are roughly balanced in the different bins, but we

do tend to have more mutable examples in the least frequent bins and more immutable in the higher-frequency bins. From the plots in Figure 6 and Figure 7 we see that the performance differs across bins, we obtain high accuracy (>90%) in lower and upper bins, while mid bins accuracy is around 85% for the Imm-1 classifier and around 80% for the Imm-N classifier (it differs somewhat per model). However, we note that the performance of the classifiers cannot be explained only by frequency. For the Imm-N classifier e.g. in the 3rd bin, there are 609 immutable and 1052 mutable examples. Depending on the model, the classifier obtains between 70-95% accuracy, whereas a random classi-

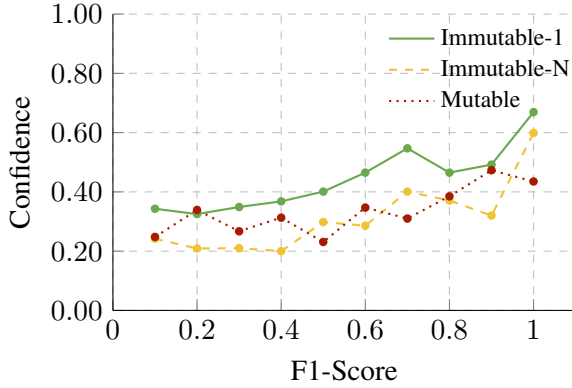


Figure 5: Average confidence per F1-Score for Falcon over all the queries in M_{ULAN}.

fier using only the frequency would obtain 60%. The same goes with a higher bin, where there are 995 immutable and 639 mutable examples, and the accuracy is still ranging between 65-85% whereas a classifier using only the frequency would obtain 55%. In the same line for the Imm-1 classifier, we see that in the 3rd bin there are 964 immutable-1 and 840 mutable examples, and the models get around 85-95% accuracy, way above a classifier using simple frequency (where low frequent examples would be considered mutable, therefore getting 46% accuracy). Similarly, in higher bins (9th), there are 1071 and 818 immutable-1 and mutable examples respectively, and the classifiers obtain 75-85% accuracy, indicating that they are not simply considering all these high frequent examples mutable.

E MDL Experimental Setup

We train the probes using the Adam optimizer (Kingma and Ba, 2014), with weight decay 0.01 and the default learning rate in the HuggingFace trainer (5e-5). We use a linear learning rate scheduler with warm-up ratio of {0.0, 0.1, 0.2}.¹⁷ We use early stopping with patience 4 and we train for a maximum of 12 epochs.

F Updates Data

After the selection process described in the Edits (Q3) section we obtain the following number of examples per model: 177 (LLaMA), 406 (Alpaca), 182 (LLama-2), 2138 (LLama-2_{Chat}), 136 (Falcon), 583 (Falcon_{Instr}).

¹⁷Defined specifically for each probe by using the one that gives the best validation accuracy when the probe is trained on all the data (on t_S).

		Models					
Relation		Llama		LLaMA-2		Falcon	
Immutable	P19	39.57	31.01	53.94	38.41	37.28	20.91
	P20	15.56	36.93	29.67	45.02	7.64	15.75
	P30	92.31	92.31	92.73	88.39	88.91	85.60
	P36	61.57	57.35	76.43	63.15	58.99	59.41
	P103	87.25	85.73	86.55	78.55	83.68	84.11
	P138	34.82	27.37	35.50	44.51	26.81	16.71
	P140	41.55	42.50	41.88	43.30	39.85	44.04
	P159	52.21	58.69	62.18	63.66	55.13	53.72
	P364	84.56	76.98	86.10	58.15	88.82	80.85
	P449	52.34	41.61	52.27	49.28	36.46	34.25
P495	64.50	44.33	55.27	53.69	52.40	51.75	
P740	34.54	42.08	36.09	50.53	34.49	31.39	
Average		55.07	53.08	59.05	56.39	50.87	48.21
Immutable-N	P27	79.85	77.69	79.58	80.05	78.02	76.35
	P47	30.20	32.40	30.49	30.32	27.60	24.34
	P69	52.47	54.04	53.38	48.19	47.20	43.15
	P101	30.96	43.90	37.14	55.49	22.50	31.59
	P136	43.63	47.61	44.79	48.55	45.76	39.78
	P166	35.07	37.34	34.74	33.81	29.50	30.14
	P190	28.05	16.12	27.84	8.69	27.58	26.34
	P530	66.39	60.96	70.83	73.24	61.76	64.92
	P1303	38.54	40.61	37.85	41.39	39.37	30.00
	P1412	79.37	88.39	89.80	89.11	72.75	85.56
Average		48.45	49.90	50.65	50.88	45.21	45.22
Mutable	P6	1.73	4.25	2.56	10.51	1.73	5.30
	P39	55.56	55.34	58.48	61.05	44.56	46.53
	P54	32.59	48.52	53.62	42.53	31.85	45.14
	P108	38.45	45.25	43.38	42.91	40.60	38.29
	P210	7.01	13.95	7.18	11.69	1.81	8.94
	P264	45.75	45.90	42.28	42.11	15.90	30.38
	P286	14.88	24.60	23.11	24.10	11.13	17.14
	P451	24.08	21.97	21.04	23.59	12.98	14.50
	P488	7.71	22.57	6.85	21.25	3.88	13.08
	P551	26.52	35.73	23.87	36.03	24.48	31.13
P937	47.10	49.17	30.01	50.51	16.72	38.72	
P1037	12.94	18.68	18.51	17.25	14.16	11.43	
P1308	5.78	27.46	14.96	33.89	4.43	11.33	
Average		24.62	31.80	26.60	32.11	17.25	23.99
mAverage		41.94	44.32	44.66	45.85	36.84	38.42

Table 7: F1-scores of each foundation model (first column) and their instruction counterpart (second) on every relation in M_{ULAN}, broken down by query type i.e., Immutable-1, Immutable-N and Mutable and the overall Macro Average.

Percentile	Imm-N		Imm-1	
	Immutable-N	Mutable	Immutable-1	Mutable
0	8	1414	350	1414
1	408	1262	840	1167
2	609	1052	964	840
3	439	1338	831	1262
4	379	1330	859	1236
5	519	1078	732	994
6	804	930	856	1140
7	918	729	1051	860
8	995	639	1071	818
9	1153	498	1341	539

Table 8: Examples counts per frequency bin for each probe classifier.

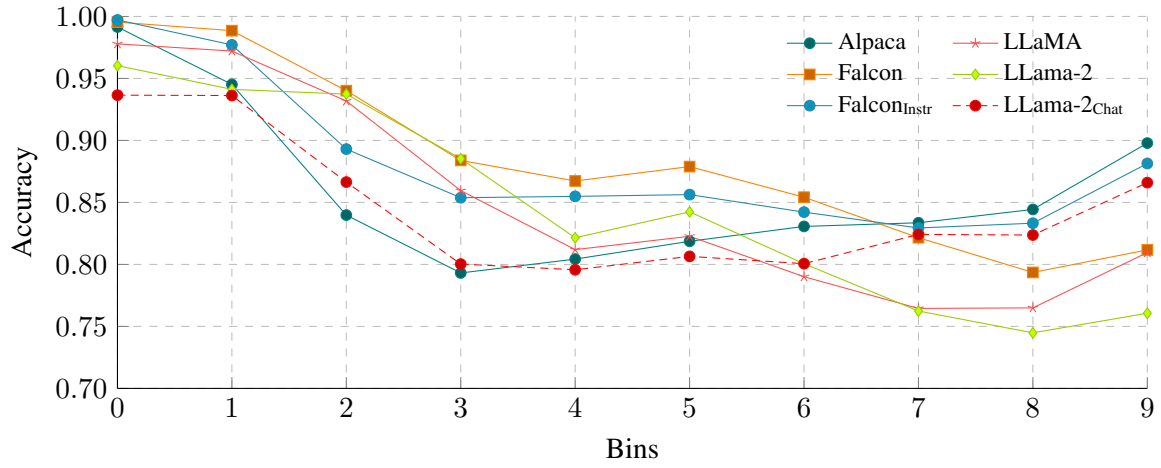


Figure 6: Accuracy of the MDL classifier per percentile for all six models on the **Immutable-1** types of relations.

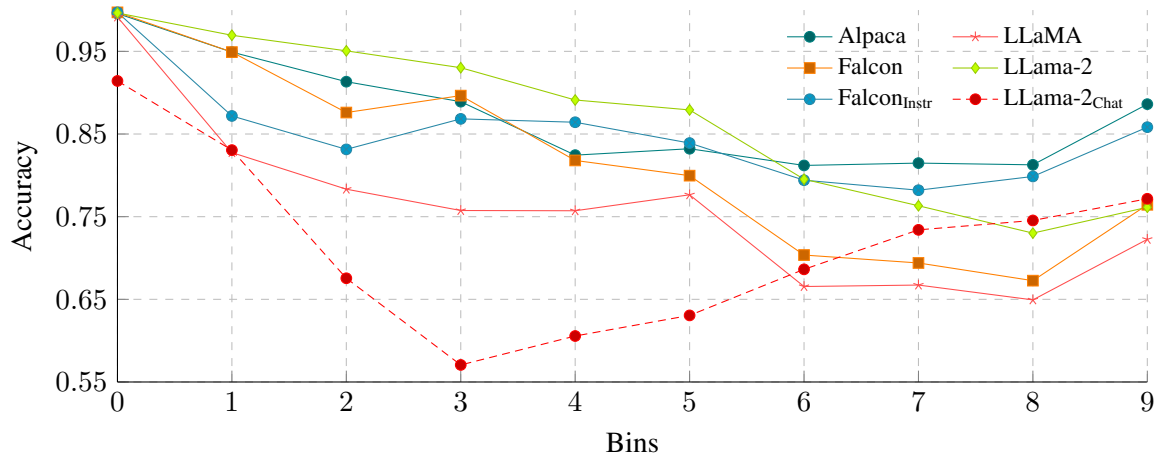


Figure 7: Accuracy of the MDL classifier per percentile for all six models on the **Immutable-N** types of relations.