Unified Lookup Tables: Privacy-Preserving Foundation Models

Nikita Janakarajan[1,2*] **Irina Espejo Morales**[1*] **Marvin Alberts**[1,3] **Andrea Giovannini**[1]
Matteo Manica[1] **Antonio Foncubierta-Rodríguez**[3]
[1]IBM Research Europe [2]ETH Zürich [3]University of Zurich
∗Equal contribution
{nja,agv,tte,fra}@zurich.ibm.com
{Irina.Espejo.Morales,Marvin.Alberts}@ibm.com

Abstract

Transformers, despite their success in a variety of sequence modeling tasks, have a significant limitation: they are inherently data-greedy, which can lead to overfitting when the data are scarce. In such cases, common practice is to build a Foundation Model (FM), a model trained on large amounts of publicly available data, that can then be fine tuned for a specific task. Another known problem of FMs is training data leakage. It has been demonstrated that excerpts of the training data can be obtained by prompt engineering on a FM, which poses a high risk of exposing confidential data. In this paper we propose Unified Lookup Tables (ULTs), a data preprocessing step for building and fine tuning a FMs in a privacy preserving manner, which simultaneously enables the reuse of a trained model on new datasets without exposing any training data. The method relies on data compression methods as efficient modality tokenizers, and a common representation vocabulary for all datasets. We evaluate the effect of using ULT with a text compression mechanism in training both decoder-only and encoder-decoder language models. Results show that the evaluation loss decreases consistently compared with the raw data one when using ULT on different data domains, proving that the transformation does not negatively affect model training. Moreover, we evaluate the performance of adopting ULT on natural language showing that the resulting model exhibits an average relative increase of $\sim$16% on a collection of text metrics. The experiments using ULT as pre-processing step with chemical reaction data on the task of forward prediction report non-significant performance degradation with respect to training with traditional SMILES strings. Finally, we perform experiments to test the privacy preserving capabilities of the ULT pre-processing on a realistic setup of data consisting of chemical reactions, a field in which confidentiality is a key factor and a major intellectual property concern. Our series of experiments show that ULTs are attack-proof without the entire dataset from which it is built. Even if partially correct mappings of the ULT are generated by mixing datasets, an attacker cannot successfully decode the data. Code to reproduce the experiments is available at: https://ibm.biz/unified-luts.

1 Introduction

In recent years, we have witnessed a revolution in the field of Artificial Intelligence (AI). On the application side, AI models are being applied to multiple disciplines and releasing Large Language Models (LLMs) to the general public. [22] has reinforced the idea that large AI models can do anything. On the technical side, AI has evolved from feature-engineered systems, to architectures that incorporate inductive bias like Convolutional Neural Networks (CNNs) and ResNets [10], to Trans-

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

fomer architectures in which data completely drive the learning through attention mechanisms [24]. There are, however, requirements for these architectures to succeed, namely, extensive amounts of data and compute. Challenges related to compute can be overcome to some extent. One can increase the number of compute nodes, invest in making nodes faster, more efficient, or both. However, challenges with respect to data persist. Data scarcity, in addition to concerns about data governance, ownership and training data leakage, prevent more than a few application domains from benefiting from the AI revolution. When data are scarce, large models are susceptible to the risk of over-fitting and failure to generalize beyond the training data. A common solution is to first train these, so called, foundation models on large datasets somewhat informative to the task at hand and fine-tune them on a domain-specific dataset. While this circumvents issues relating to data scarcity, it imposes a need for such large general datasets to be broadly available in the first place. Moreover, fine-tuning on a niche dataset, which may be highly confidential, requires that the risk associated with exposing the data be minimised or better, eliminated. With recent research proving that it is possible to prompt models in ways that expose parts of the training data [4, 11, 13, 8], it is imperative to account for these aspects when deploying foundation models in production, especially in industries where privacy and confidentiality are critical, e.g., pharmaceutical industry, banking or customer care. To address these challenges, we propose a method that relies on Unified Lookup Tables to create an encoded representation for any dataset, allowing FMs to be trained on any large, available, general dataset, while still preserving information without leakage and retaining model performance. Herein, we evaluate our methodology showing how we can train different family of models in an application-agnostic context by validating the approach on natural language and chemical reactions data. Firstly, we show that training on encoded data does not alter the learning process and increases or does not affect model performance. Finally, focusing on the case of chemical reactions, we demonstrate how it is not possible to recover data or to induce leakage by conducting a series of ablation studies when considering the case of an attacker accessing the model predictions using a large public data and increasing fractions of sensitive data.

2 Related work

A long pursued goal in AI is to train models that can ingest any type of data without architectural changes. However, for a long time this has not been possible due to the limitations of the architectures: the most obvious limitation is that a fixed number of neurons at the input or output layers impose restrictions on the dimensionality of the data, but even if this problem was avoided, the dynamic range of the data (in case of numbers) is a similar limitation. Sequence to sequence models have the advantage that the data is ingested sequentially, without a fixed length, but the models do not extend from one dataset to another if they use different vocabularies. With the intention of ingesting multiple data types, various efforts have tried to directly ingest the data as byte sequences [14, 28]. Jaegle et al. propose the *Perceiver* architecture, which uses cross attention to map a byte array and a latent array to a fixed dimension bottleneck, followed by a transformer. Yu et al. go further and propose an architecture based on multiscale transformers that can autoregressively predict million byte sequences (hence the name *Megabyte*) Both models are able to ingest any type of data, but because there is no unified representation for the byte arrays, a model trained on a dataset cannot be transferred to another dataset, since the learnt patterns will refer to the raw data in absolute terms. E.g., byte arrays for the same image are different whether they are stored as JPEG or PNG; byte arrays for textual data are different depending on the encoding used. This ties each model to the representation used for the training data. In this paper we propose a method that decouples the model from the representation, allowing transfer from one dataset to another. This opens the possibility to train models on data that until now was not AI-ready, because of lack of large-scale normalization. However, as discussed before, the privacy and confidentiality issue remain. Inan et al. have analyzed the data leakage problem in LLM, which happens because large models have the ability to memorize rare training data samples. This is a particularly problematic threat for domain-specific models, those whose intended users are not the general public, but individuals or companies that own sensitive information on which they would like to use AI, without the risk of exposing the data through a model that can be attacked to reproduce pieces of its training data. Multiple efforts have been made to preserve privacy, for instance, by generating synthetic data based on the data that needs to be protected [27, 29]. Or through federation when multiple stakeholders want to train a model together without exposing the data to each other [9]. Our method proposes to encode the data before expos-

ing them to the model, and keep the encoding/decoding steps separate from the deployment of the model, allowing the model to be shared and iteratively improved, without losing control of the data.

3 Methods

In this section we describe our method, called Unified Lookup Tables. The goals of the method are to enable any dataset, of any data type, to be ingested by language models and in a way that allows for training and, potentially, domain transfer, without directly exposing confidential data to the model. ULT is simply a pre-processing step that encodes the data before being fed to the model. This step is inspired by Shannon's source coding theorem [21] and its application to optimally encoding data: symbols are assigned to *chunks* of data (patterns) such that the shortest symbols are reserved for the most frequent patterns. This ensures that the average total sequence length will be reduced, since shorter symbols will be used for encoding patterns in the data more often than the longer symbols. Knowing that transformers have a bottleneck on sequence length and on vocabulary size [5], this method of encoding data has a positive effect on training transformers. Since the mapping of the patterns to symbols depends on how frequently these patterns appear in a training corpus, a data transformation that balances the trade-off between vocabulary size and sequence length is desirable. These transformations are basically what compression methods do - reversible transformations on the data such that the content is represented as the combination of a few rare patterns and many highly frequent patterns. The mapping or Lookup Table (LUT) is computed on a training corpus. This enables the privacy preserving advantage for training models on a corpus of confidential data. If the confidential corpus has a similar distribution as public corpora, it becomes unnecessary to train on the confidential data because there is no domain shift. If, in contrast, the confidential corpus and a public corpus have different pattern distributions, the symbols will be assigned to patterns differently, and therefore one LUT cannot be used to decode data that were encoded with another LUT. However, we hypothesize that this does not hinder the training of a Language Model (LM), to which symbols retain the same meaning across datasets, that is, the n-th symbol represents the n-th most frequent pattern in the data, even if the data come from different domains. Figure 1 shows an overview of the encoding and decoding process, and Figure 4 in the Appendix depicts how this preserves privacy during training and inference of the model.

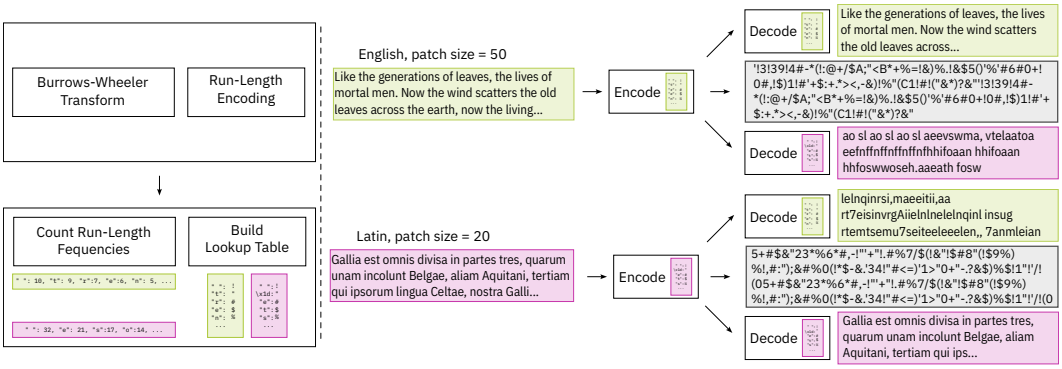


Figure 1: Encoding and decoding mechanism with ULT. On the left, the offline process that computes the LUT for a training set. On the right, two examples of encoding and decoding with ULT. The green training data is used to generate a lookup table with compressed patches of data. The same is done with another training set, shown in pink color. Decoding is only successful when the lookup table corresponds to the one used during encoding.

Compression technique

The Burrows-Wheeler Transform (BWT) or block sorting transform [3], when applied to a string of text, permutes characters in a reversible way, making it more probable to find repeating characters, and thus, easier to compress. This property has been exploited by well known methods like the GZip compressor. In [18], various methods of compression based on BWT are discussed, one of them being the combination of BWT and Run Length Encoding (RLE). We opt for this method as a

generic compression mechanism for ULT. We divide the input sequences in patches, and transform each patch using BWT. This text is then encoded using RLE, so that we also encode the number of times a character is repeated. For examples on compressing a single sentence with different patch sizes, see Appendix B.

Unified Lookup Tables

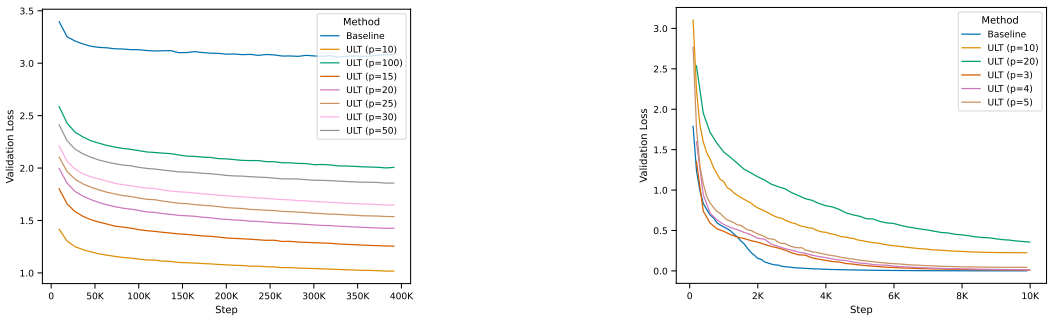
In order to unify the representation of data for language models, we propose to map all data to the same vocabulary. Without loss of generality, we use Unicode[1] characters as symbols, because they are easy to render for demonstration and supported by most language models available, but also because the way that Unicode characters are sorted is loosely based on the key concept of our approach, with the more frequent characters being assigned shorter codes than the less frequent characters. Beyond the compression benefit, keeping the LUTs sorted by frequency is what guarantees that the models are reusable, since the relative meaning of each symbol is constant across datasets.

4 Results

We define a set of experiments to verify our hypotheses that a compression-based encoding: (i) does not hinder learning; (ii) when tuned does not deteriorate performance; (iii) guarantees privacy is preserved through the encoding/decoding step as long as the LUTs are kept private. To validate the first two hypotheses, we train a decoder-only architecture on natural language and an encoder-decoder architecture on chemical reaction data to evaluate their performance. For the third hypothesis, we design a decoder attack experiment based on ULTs pre-processing to quantify the privacy guarantees when using a public chemistry reactions dataset to decode a smaller private reactions dataset. All experiments are detailed in Appendix A.

ULT improves over baseline in natural language

We evaluate the application of ULT at varying patch sizes in comparison with a baseline relying on the same model without applying any data encoding focusing on two aspects: (i) loss behaviour over training (ii) model performance dependency on the data encoding strategy. Figure 2a reports the evolution of the validation loss comparing the behaviour of models trained on non-encoded text (Baseline) and various ULT-encoded data showing how, besides a constant offset due to different sequence lengths and vocabulary sizes, the encoding process does not alter loss trends. A similar observation is made for training loss as seen in Appendix Figure 5b.



(a) Evolution of validation loss on Wikitext.

(b) Evolution of validation loss on USPTO.

Figure 2: Analysis of model training using ULT. Evolution of the validation loss during model training on (a) Wikitext and (b) USPTO varying the patch size (p) in comparison with a baseline trained on non-encoded natural language and non-encoded SMILES strings, respectively.

Table 1 reports a series of metrics, i.e., ROUGE (varying n-grams and sub-sequences), Meteor and BERT-F1 for all ULT variations and the baseline, highlighting how ULT-based models either outperforms or are competitive with the model trained on plain text. Especially looking at the results for patch size 10, we observe an average relative improvement of $\sim$16% over the Baseline.

[1] in UTF-8 encoding form, https://home.unicode.org/

	ROUGE1	ROUGE2	ROUGEL	ROUGELSUM	METEOR	BERT-F1
Baseline	0.29	0.16	0.26	0.28	**0.32**	**0.85**
ULT (p=10)	**0.33**	**0.23**	**0.33**	**0.33**	**0.32**	**0.83**
ULT (p=15)	**0.32**	**0.20**	**0.32**	**0.32**	0.31	0.82
ULT (p=20)	0.28	0.18	0.28	0.28	0.29	0.81
ULT (p=25)	0.23	0.18	0.23	0.23	0.27	0.80
ULT (p=30)	0.19	0.14	0.19	0.19	0.26	0.79
ULT (p=50)	0.12	0.09	0.12	0.12	0.21	0.79
ULT (p=100)	0.01	0.00	0.01	0.01	0.05	0.75

Table 1: Comparison of ROUGE (F1-measure for different n-grams and longest common subsequences), METEOR, and BERT-F1 scores on the test set for a model trained without ULT (Baseline) and with ULT at various patch sizes (p). In bold we report the two best performing methods.

	ACCURACY TOP-1	ACCURACY TOP-2	ACCURACY TOP-5	ACCURACY TOP-10
Baseline	**0.78**	**0.83**	**0.87**	**0.89**
ULT (p=3)	0.69	0.72	0.75	0.76
ULT (p=4)	**0.69**	**0.73**	**0.76**	**0.77**
ULT (p=5)	0.44	0.51	0.58	0.62
ULT (p=10)	0.04	0.06	0.08	0.11
ULT (p=20)	0.03	0.05	0.08	0.09

Table 2: Comparison of top 1, 2, 5 and 10 accuracies for product generation on the test set of USPTO for models trained without ULT (Baseline) and with ULT at various patch sizes (p). In bold we report the two best performing methods.

ULT achieves competitive results on chemical reactions data

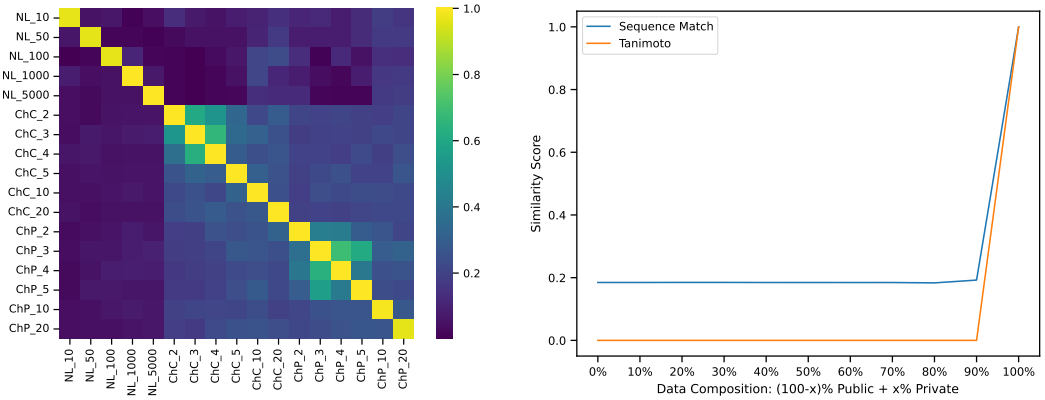
First, we compare the training and evaluation loss during pre-training on USPTO without ULT (canonicalized raw SMILES strings) and with ULT pre-processing (compressed SMILES) for different patch sizes. The validation loss curves in Figure 2b of models trained with ULT display comparable trends as the baseline model. The choice of the patch size has an expected effect on the saturating loss of validation data. The training loss curves shown in Appendix 6b show a similar behaviour. In contrast to natural language, SMILES strings are very homogeneous (most of SMILES strings contain the character "C" for carbon), it is short-length patterns that capture the nature of the molecule. Consequently, if the patch size increases then two different patches become more similar, in other words, if the patch is too big the signal (short-length patterns) is averaged out within the compressed patch. This accounts for the decrease in performance observed in the model with an increase of patch size. Secondly, we evaluate the performance of models trained with ULT on forward reaction prediction, i.e., generation of products given reagents. We compare this performance against the baseline, as shown in Table 2. We find that with an optimal patch size, in this case $p = 4$, the model trained with the ULT shows a competitive performance against the baseline. Baseline and ULT experiments used the same hyper-parameters, which were optimized for the training the baseline model. The accuracy metric we report is calculated by exact full string match of the generated products with respect to the ground-truth.

ULT preserves the privacy of the training data

Blind decoding of products from the confidential dataset fails if the appropriate LUT is not used as seen in Figure 3a. Elements on the diagonal obtain perfect or near perfect sequence matching ratios, meaning that the data can be successfully encoded and decoded with the same LUTs. Outside the diagonal, one can see that sequence match ratios are comparatively much lower suggesting the difficulty in retrieving the products without the appropriate LUT. Encoding and decoding within domain (confidential chemical reactions and public chemical reactions) appears to be less difficult than across domains (chemical reactions and natural language), as suggested by the slightly higher sequence match ratios. Furthermore, it is also seen that patch sizes strongly affects the ability to decode even on the same dataset. The exception is when patch sizes are so large that it covers the entire chemical sequence.

We further prove that even within the same domain, the ULTs are robust to attacks without $> 90\%$ of the data it is built from as seen in Figure 3b. In other words, an attacker would need to know at least 90% of the confidential data to successfully decode the rest. In a similar experiment, we show

that even by operating on the LUTs directly and increasing the percent of correct mappings, the private, confidential data cannot be successfully decoded *without* a fully correct LUT (see Appendix Figures 7b and 7a). Similar observations are made for patch sizes 3, 5, 10 and 20 as shown in Appendix H.



(a) Similarity ratio of encoding/decoding products from the confidential dataset with different LUTs.

(b) LUTs (p=4) built from a public dataset increasingly incorporated with samples from the private dataset.

Figure 3: Comparison of similarity scores between the original and encoded/decoded sequence. a) shows the similarity score when the decoding LUT is built with different parameters or datasets versus the encoding LUT. Labels : *ChC* - confidential chemical reactions, *ChP* - public chemical reactions, *NL* - natural language, followed by patch size. b) shows the similarity score when different percentages of the data used to encode the sequence are known at decoding time.

5 Conclusion

We introduce ULT, a novel form of data pre-processing based on classical compression methods and motivated by the limitation of modern LLM architectures in preventing training data leakage. This limitation deters the development of broadly available models trained on sensitive, proprietary, or confidential datasets, like chemical reaction data in the pharmaceutical industry, and which can benefit society as a whole. Using ULT pre-processing enables model training with siloed datasets without exposing any specifics. We demonstrate that training with the ULT compressed dataset (i) does not degrade performance in the generation task compared to training without ULT; (ii) allows training on data with distinct underlying probability distributions, such as natural language and chemical reactions, without modifications to the compression method; (iii) allows training with different LLM architectures with comparable results; and (iv) protects against adversarial attacks, preventing information leakage. ULT's independence from data type makes it suitable for processing different types of text data and compressing other data modalities. A natural extension of our work lies in applying similar techniques on additional modalities, such as, images, time series or combinations of these. For instance, processing vision data using standard image compression algorithms instead of BWT-based text compression. The versatility of ULT makes learning a joint compressed representation of multiple modalities straightforward. ULT pre-processing on modalities essentially compresses them to a string, which can be fed to any string-processing model, removing the need for multiple models for each modality. Nevertheless, the ULT pre-processing method is not without its limitations. As observed in the natural language and chemical reactions experiments, selecting a sub-optimal patch size parameter results in performance degradation. We recommend that the patch size be treated as a hyperparameter to be tuned for the domain at hand. This could create a computational bottleneck in the end-to-end model-building pipeline despite the final compressed version of the data using less memory and making models train faster. Another limitation prevalent in research on privacy-preservation is that without a theoretical proof of privacy guarantees, one cannot disregard the fact that an attacker could find ingenious ways in special scenarios to retrieve part of the confidential dataset. We believe from the decoder attack experiments that the ULT pre-processing method adds a significantly challenging barrier to hacking the LUT and subsequently recovering the data, especially when handling different modalities.

References

[1] Joshua Ainslie et al. "GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 4895–4901. DOI: 10.18653/v1/2023.emnlp-main.298. URL: https://aclanthology.org/2023.emnlp-main.298.

[2] Loubna Ben Allal et al. *SmolLM - blazingly fast and remarkably powerful*. 2024.

[3] Michael Burrows and David Wheeler. *A block-sorting lossless data compression algorithm*. Tech. rep. 124. Digital Equipment Corporation, 1994.

[4] Nicholas Carlini et al. "Extracting Training Data from Large Language Models". In: *CoRR* abs/2012.07805 (2020). arXiv: 2012.07805. URL: https://arxiv.org/abs/2012.07805.

[5] Grégoire Delétang et al. "Language modeling is compression". In: *arXiv preprint arXiv:2309.10668* (2023).

[6] P Kingma Diederik. "Adam: A method for stochastic optimization". In: *(No Title)* (2014).

[7] Brenda S Ferrari et al. "Predicting polymerization reactions via transfer learning using chemical language models". In: *npj Computational Materials* 10.1 (2024), p. 119.

[8] Jonas Geiping et al. *Coercing LLMs to do and reveal (almost) anything*. 2024. arXiv: 2402.14020 [cs.LG]. URL: https://arxiv.org/abs/2402.14020.

[9] Thierry Hanser. "Federated learning for molecular discovery". In: *Current Opinion in Structural Biology* 79 (2023), p. 102545.

[10] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[11] Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. *Are Large Pre-Trained Language Models Leaking Your Personal Information?* 2022. arXiv: 2205.12628 [cs.CL]. URL: https://arxiv.org/abs/2205.12628.

[12] Huseyin A Inan et al. "Training data leakage analysis in language models". In: *arXiv preprint arXiv:2101.05405* (2021).

[13] Daphne Ippolito et al. *Preventing Verbatim Memorization in Language Models Gives a False Sense of Privacy*. 2023. arXiv: 2210.17546 [cs.LG]. URL: https://arxiv.org/abs/2210.17546.

[14] Andrew Jaegle et al. "Perceiver: General perception with iterative attention". In: *International conference on machine learning*. PMLR. 2021, pp. 4651–4664.

[15] Wengong Jin et al. "Predicting organic reaction outcomes with weisfeiler-lehman network". In: *Advances in neural information processing systems* 30 (2017).

[16] Greg Landrum. *Open-source cheminformatics software*. 2024. URL: https://www.rdkit.org/.

[17] The Python Standard Library. *difflib package*. 2024. URL: https://docs.python.org/3/library/difflib.html.

[18] Giovanni Manzini. "An analysis of the Burrows—Wheeler transform". In: *Journal of the ACM (JACM)* 48.3 (2001), pp. 407–430.

[19] Stephen Merity et al. *Pointer Sentinel Mixture Models*. 2016. arXiv: 1609.07843 [cs.CL].

[20] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140 (2020), pp. 1–67.

[21] Claude E Shannon. "Coding theorems for a discrete source with a fidelity criterion". In: *IRE Nat. Conv. Rec* 4.142-163 (1959), p. 1.

[22] Irene Solaiman et al. *Release Strategies and the Social Impacts of Language Models*. 2019. arXiv: 1908.09203 [cs.CL]. URL: https://arxiv.org/abs/1908.09203.

[23] T. T. Tanimoto. *An Elementary Mathematical theory of Classification and Prediction*. Tech. rep. IBM, 1958.

[24] Ashish Vaswani et al. "Attention is all you need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[25] David Weininger. "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules". In: *J. Chem. Inf. Comput. Sci.* 28 (1988), pp. 31–36. URL: https://api.semanticscholar.org/CorpusID:5445756.

[26] Thomas Wolf et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Ed. by Qun Liu and David Schlangen. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. URL: https://aclanthology.org/2020.emnlp-demos.6.

[27] Da Yu et al. “Privacy-Preserving Instructions for Aligning Large Language Models”. In: *arXiv preprint arXiv:2402.13659* (2024).

[28] Lili Yu et al. “MEGABYTE: Predicting Million-byte Sequences with Multiscale Transformers”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 78808–78823. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/f8f78f8043f35890181a824e53a57134-Paper-Conference.pdf.

[29] Ziqian Zeng et al. “PrivacyRestore: Privacy-Preserving Inference in Large Language Models via Privacy Removal and Restoration”. In: *arXiv preprint arXiv:2406.01394* (2024).

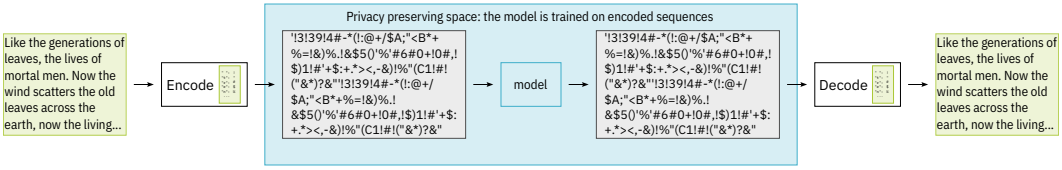


Figure 4: The data is encoded prior to being fed to the model, and because the model does not contain information about the LUT, the model only learns the relative patterns in the data, preserving privacy.

A Experiments

We have designed a set of experiments to verify our hypotheses that a compression-based encoding: (i) does not hinder learning; (ii) when tuned does not deteriorate performance; (iii) guarantees privacy is preserved through the encoding/decoding step as long as the LUTs are kept private. To validate the first two hypotheses, we train a decoder-only architecture on natural language and an encoder-decoder architecture on chemical reaction data to evaluate their performance. These experiments are detailed in the subsequent sections "Training language models on encoded data" and "Training models on encoded chemical reactions". The goal for both is to show how leveraging ULT-based encoding/decoding does not hinder the training process and that the resulting trained models, when using appropriate compression parameters, are competitive or outperform models trained directly on natural language and chemical reactions, respectively. By selecting two different architectures and text modalities for our experiments, encoder-only with natural text and encoder-decoder with chemical reactions, we also prove that the application and performance of ULTs pre-processing is generalizable to data and model architecture. For the third hypothesis, we design a decoder attack experiment based on ULTs pre-processing to quantify the privacy guarantees when using a public chemistry reactions dataset to decode a smaller private reactions dataset. For the latter decoding experiment, the choice of chemical reactions datasets serves as an example of a domain in which privacy and confidentiality are critical. The decoding attack experiment is detailed in the Experiments subsection "Decoding attack on SMILES private dataset".

All training experiments are executed on 8 NVIDIA A100-80GB GPUs using the distributed training package of the `transformers` library [26].

Training language models on encoded data

For natural language experiments, we consider a decoder-only architecture based on SmoLLM-135M [2], that relies on grouped-query attention (GQA) [1] prioritizing model depth (30 layers, 9 heads, 3 KV-heads, embedding size 576 and hidden dimension 1536) with a context length of 2048 tokens. For a detailed description of the architecture parameters see Appendix D. We train the model via causal language modeling on `wikitext-103-raw-v1` [19], a dataset comprising high-quality and featured Wikipedia articles with over 100 million tokens, using the provided splits (1.8M/3.76K/4.36K rows respectively for training, validation and test). We consider different ULT configurations training for 637140 steps using a batch size of 262144 tokens (following the settings recommended by **?**) and half-precision, selecting the best checkpoint via validation loss. All training runs use a starting learning rate of $3\mathrm{e}{-4}$, a cosine scheduler, weight decay (0.1) and Adam as an optimizer (β_1=0.9, β_2=0.95) [6]. Full details on training parameters can be found in Appendix E. For evaluation, we prompt the models using the first ten words for each test example sampling a maximum of 64 new tokens using beam-search multinomial sampling with 3 beams.

Training models on encoded chemical reactions

For the following experiments, we train an encoder-decoder architecture based on T5-small [20] (12 layers, 8 heads, embedding size 512 and feed-forward dimension 2048) with a context length of 512 on chemical reactions. For a detailed description of the architecture parameters, see Appendix D. The model is trained on the USPTO dataset [15] of chemical reactions, formatting molecules as SMILES [25]. Each reaction consists of a set of reagents and a given product, see Appendix C for

examples. The model is trained to predict the correct product given the reagents. The USPTO dataset is split into train, test and validation split with 400k, 40k and 30k samples respectively. This dataset plays the role of a public, reasonably large and well-known dataset in the chemistry community that an attacker could have access to. In all experiments we train for 10000 steps and a batch size of 131072 tokens, using a starting learning rate of $5\mathrm{e}{-4}$, cosine scheduler, weight decay (0.1) and Adam as an optimizer (β_1=0.9, β_2=0.95) [6]. Full details on training parameters can be found in Appendix E. Generation for each set of reagents in test set is performed with 15 beams, temperature 0, and by sampling 10 product candidates.

Decoding attack on SMILES private dataset

In this experiment, we evaluate the privacy preserving capabilities of our method. To this end, we use a small dataset of around 3000 polymerization reactions [7] to play the role of a *private*, domain specific and confidential dataset with limited visibility. This data is not included in the USPTO dataset, which will serve as the large, publicly available dataset. We analyze the performance of encoding and decoding with various LUTs, evaluating how much of the original data can be recovered if an attacker had access to the encoded outputs of a trained model. Additionally, we also simulate a scenario where increasing percentages of the correct LUT mappings are discovered, and identify a minimum discovery threshold – the percentage of the correct LUT required to correctly expose the confidential training data. We use sequence match ratio (as implemented in `difflib` [17]) and Tanimoto similarity [23](as implemented in `RDKit` [16]) between samples from the confidential chemical reactions dataset and the encoded-decoded versions as a generic similarity metric. We perform multiple experiments, ranging from the ones in which the attacker has no knowledge about the data, the ULT patch size or even the domain, to the ones in which the attacker has information – either about the data or some of the exact mappings.

Decoding blindly

In this experiment, we assume that an attacker does not know on which data the model has been trained, and that somehow, they gain access to the encoded training set. The attacker has no information about the patch size used for compression and the ULT mappings. To evaluate this setting, we encode and decode a dataset with all combinations of pairs of LUTs generated in the previous experiments to verify if an attacker can successfully decode the data with LUTs generated from other sources. Reagents and products are encoded with LUTs built on the reagents and products in the training set, respectively.

Decoding with partial information about the data

In this experiment, we assume that an attacker has access to a public dataset from the same domain of the private data, and gains access to an increasing amount of samples from the private data. The aim is to estimate how much of the private data needs to be leaked for the attacker to successfully determine the LUT to decode the entirety of it. Reagents and products are encoded with LUTs built on the reagents and products in training set, respectively. We generate LUTs for the mixed public-private datasets and estimate this required leakage threshold of private data by measuring the sequence match ratio and Tanimoto similarity of the decoded reagents and products with their groundtruth sequences.

Decoding with partial information about the lookup table

In the final experiment of the attacker series to prove our method is attack-proof, we simulate the scenario of the attacker correctly identifying some of the LUT mappings. We test this scenario in two settings - increased correct mappings in order of frequency and randomly, where each increased percent is a super set of the preceding percentages.

Patch size	$RLE(BWT(s))$	Compression ratio
10	e*rmsrafTont,s [...]	0.94:1
20	t,s*r mdsprafsT[...]	1.0:1
50	on,syreas* rvuc[...]	1.1:1
100	ien2,steyfargea[...]	1.1:1
200	n.enyahey2,sen[...]	1.2:1
500	c.af2.2nd,erney[...]	1.4:1
1000	o.3af2.d4np.edn[...]	1.5:1
2000	2fs.3af2.d4n.d.[...]	1.6:1

Table 3: Example of the BWT-RLE text compression method applied to the abstract of this paper with different patch sizes. The end of patch is represented through a * sign.

B BWT-RLE compression examples

C USPTO dataset examples

Atom mapping before canonicalization

```
[CH2:15]([CH:16]([CH3:17])[CH3:18])[Mg+:19].[CH2:20]1[O:21][CH2:22][CH2:23][CH2:24]1.[Cl-:14].[OH:1][c:2]1[n:3][cH:4][c:5]([C:6](=[O:7])[N:8]([O:9][CH3:10])[CH3:11])[cH:12][cH:13]1>>[OH:1][c:2]1[n:3][cH:4][c:5]([C:6](=[O:7])[CH2:15][CH:16]([CH3:17])[CH3:18])[cH:12][cH:13]1
[CH3:14][NH2:15].[N+:1](=[O:2])([O-:3])[c:4]1[cH:5][c:6]([C:7](=[O:8])[OH:9])[cH:10][cH:11][c:12]1[Cl:13].[OH2:16]>>[N+:1](=[O:2])([O-:3])[c:4]1[cH:5][c:6]([C:7](=[O:8])[OH:9])[cH:10][cH:11][c:12]1[NH:15][CH3:14]
[CH2:1]([CH3:2])[n:3]1[cH:4][c:5]([C:22](=[O:23])[OH:24])[c:6](=[O:21])[c:7]2[cH:8][c:9]([F:20])[c:10](-[c:13]3[cH:14][cH:15][c:16]([NH2:19])[cH:17][cH:18]3)[cH:11][c:12]12.[CH:25](=[O:26])[OH:27]>>[CH2:1]([CH3:2])[n:3]1[cH:4][c:5]([C:22](=[O:23])[OH:24])[c:6](=[O:21])[c:7]2[cH:8][c:9]([F:20])[c:10](-[c:13]3[cH:14][cH:15][c:16]([NH:19][CH:25]=[O:26])[cH:17][cH:18]3)[cH:11][c:12]12
```

After canonicalization

```
C1CCOC1.CC(C)C[Mg+].CON(C)C(=O)c1ccc(O)nc1.[Cl-]>>CC(C)CC(=O)c1ccc(O)nc1
CN.O.O=C(O)c1ccc(Cl)c([N+](=O)[O-])c1>>CNc1ccc(C(=O)O)cc1[N+](=O)[O-]
CCn1cc(C(=O)O)c(=O)c2cc(F)c(-c3ccc(N)cc3)cc21.O=CO>>CCn1cc(C(=O)O)c(=O)c2cc(F)c(-c3ccc(NC=O)cc3)cc21
```

D Model architecture details

Decoder-only model for natural language

```
{
  "architectures": [
    "LlamaForCausalLM"
  ],
  "attention_bias": false,
  "attention_dropout": 0.0,
  "bos_token": "<|endoftext|>",
  "bos_token_id": 0,
  "eos_token": "<|endoftext|>",
  "eos_token_id": 0,
  "hidden_act": "silu",
```

```
  "hidden_size": 576,
  "initializer_range": 0.02,
  "intermediate_size": 1536,
  "max_position_embeddings": 2048,
  "mlp_bias": false,
  "model_type": "llama",
  "num_attention_heads": 9,
  "num_hidden_layers": 30,
  "num_key_value_heads": 3,
  "pad_token": "<PAD>",
  "pretraining_tp": 1,
  "rms_norm_eps": 1e-05,
  "rope_scaling": null,
  "rope_theta": 10000.0,
  "tie_word_embeddings": true,
  "torch_dtype": "float32",
  "transformers_version": "4.42.3",
  "use_cache": false,
  "vocab_size": 49153
}
```


Encoder-decoder model for chemical reactions


```
{
  "architectures": [
    "T5ForConditionalGeneration"
  ],
  "bos_token": null,
  "classifier_dropout": 0.0,
  "d_ff": 2048,
  "d_kv": 64,
  "d_model": 512,
  "decoder_start_token_id": 0,
  "dense_act_fn": "relu",
  "dropout_rate": 0.1,
  "eos_token": "</s>",
  "eos_token_id": 1,
  "feed_forward_proj": "relu",
  "initializer_factor": 1.0,
  "is_encoder_decoder": true,
  "is_gated_act": false,
  "layer_norm_epsilon": 1e-06,
  "model_type": "t5",
  "n_positions": 512,
  "num_decoder_layers": 6,
  "num_heads": 8,
  "num_layers": 6,
  "output_past": true,
  "pad_token": "<pad>",
  "pad_token_id": 0,
  "relative_attention_max_distance": 128,
  "relative_attention_num_buckets": 32,
  "torch_dtype": "float32",
  "transformers_version": "4.42.3",
  "use_cache": false,
  "vocab_size": 32100
}
```

E Training details

Trainer arguments for natural language experiments

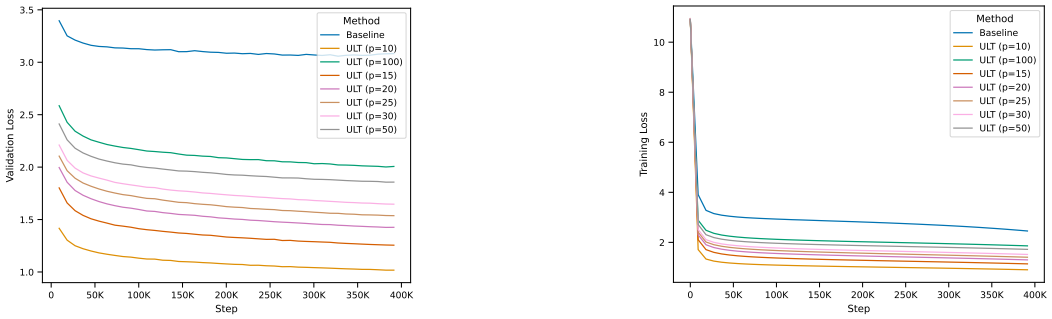
```
{
  "num_train_epochs": 70
  "evaluation_strategy": epoch
  "learning_rate": 0.0003
  "per_device_train_batch_size": 16
  "per_device_eval_batch_size": 16
  "logging_strategy": epoch
  "logging_first_step": true
  "logging_steps": 1
  "save_total_limit": 3
  "disable_tqdm": false
  "adam_beta1": 0.9
  "adam_beta2": 0.95
  "lr_scheduler_type": cosine
  "gradient_accumulation_steps": 1
  "auto_find_batch_size": true
  "weight_decay": 0.1
  "fp16": true
  "max_grad_norm": 1.0
  "dataloader_pin_memory": true
  "dataloader_num_workers": 8
  "report_to": none
  "save_strategy": epoch
  "save_steps": 1
  "metric_for_best_model": eval_loss
  "greater_is_better": false
  "load_best_model_at_end": true
}
```

Trainer arguments for chemical reaction experiments

```
{
  "num_train_epochs": 60
  "evaluation_strategy": epoch
  "learning_rate": 0.0005
  "per_device_train_batch_size": 128
  "per_device_eval_batch_size": 64
  "logging_strategy": epoch
  "logging_first_step": true
  "logging_steps": 1
  "save_total_limit": 3
  "disable_tqdm": true
  "adam_beta1": 0.9
  "adam_beta2": 0.95
  "lr_scheduler_type": cosine
  "gradient_accumulation_steps": 4
  "weight_decay": 0.1
  "fp16": true
  "max_grad_norm": 1.0
  "dataloader_pin_memory": true
  "dataloader_num_workers": 8
  "report_to": none
  "save_strategy": epoch
  "save_steps": 1
  "metric_for_best_model": eval_loss
  "greater_is_better": false
```

```
  "load_best_model_at_end": true
}
```

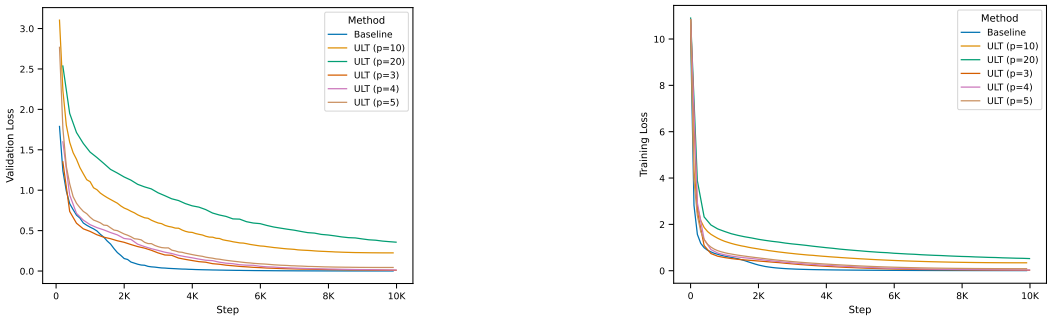
F Evolution of Loss During Model Training



(a) Evolution of the validation loss.

(b) Evolution of the training loss.

Figure 5: Analysis of model training for Wikitext using ULT. Evolution of the loss during model training on Wikitext varying the patch size (p) in comparison with a baseline trained on non-encoded natural language.



(a) Evolution of the validation loss.

(b) Evolution of the training loss.

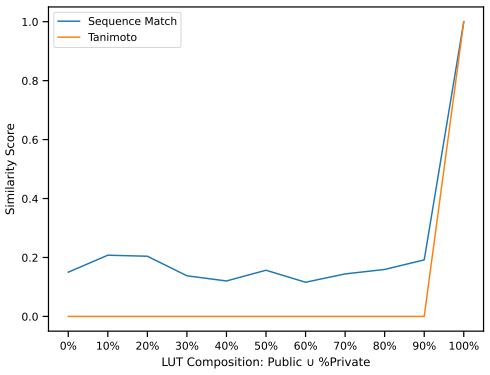
Figure 6: Analysis of model training for USPTO using ULT. Evolution of the loss during model training on USPTO varying the patch size (p) in comparison with a baseline trained on non-encoded SMILES strings.

G Natural Language Results

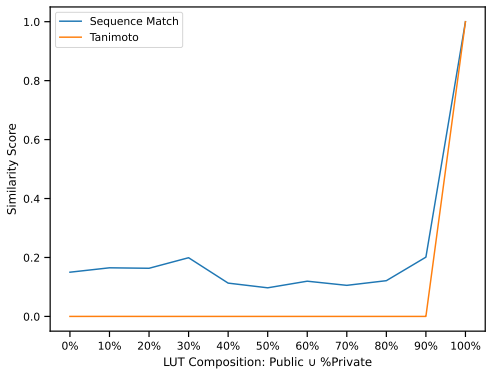
	ROUGE1	ROUGE2	ROUGEL	ROUGELSUM	METEOR	BERT-PRECISION	BERT-RECALL
Baseline	0.29	0.16	0.26	0.28	0.32	0.83	0.86
ULT (p=5)	0.36	0.24	0.35	0.35	0.35	0.85	0.85
ULT (p=10)	0.33	0.23	0.33	0.33	0.32	0.83	0.84
ULT (p=15)	0.32	0.20	0.32	0.32	0.31	0.80	0.84
ULT (p=20)	0.28	0.18	0.28	0.28	0.29	0.79	0.83
ULT (p=25)	0.23	0.18	0.23	0.23	0.27	0.78	0.83
ULT (p=30)	0.19	0.14	0.19	0.19	0.26	0.77	0.82
ULT (p=50)	0.12	0.09	0.12	0.12	0.21	0.77	0.81
ULT (p=100)	0.01	0.00	0.01	0.01	0.05	0.72	0.78

Table 4: Comparison of ROUGE, METEOR, and BERT scores on the test set for a model trained without ULT (Baseline) and with ULT at various patch sizes (p).

H Extended Results on Privacy Preservation

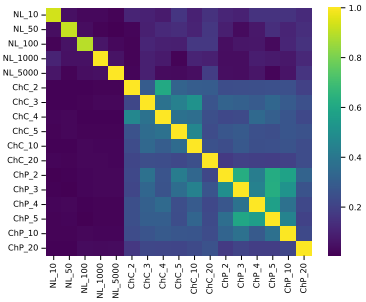


(a) LUTs built from public data and increasingly updated with correct mappings in a random order.

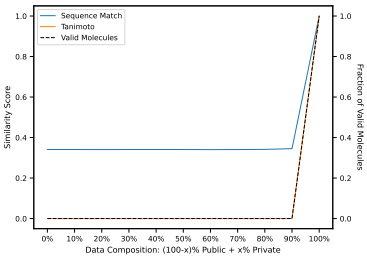


(b) LUTs built from public data and increasingly updated with the correct, frequency ordered mappings.

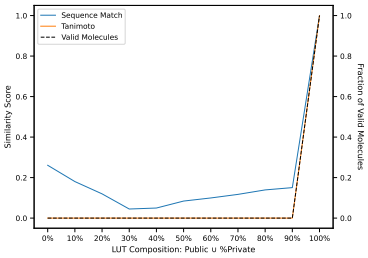
Figure 7: Similarity scores of products from the confidential chemical reactions dataset encoded with the correct LUT built from this dataset and decoded with imperfect LUTs. Patch size set to 4. Metrics are sequence match ratio and Tanimoto similarity of the decoded product and groundtruth.



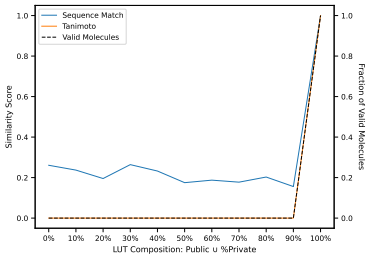
(a) Pairwise similarity ratio of reagents from the confidential chemical reactions dataset encoded and decoded with various pairs of LUTs. Labels show the dataset (*ChC* for confidential chemical reactions, *ChP* for public chemical reactions, *NL* for natural language), and patch size used.



(b) Similarity scores for reagents encoded with a LUT built from the private training set and decoded with LUTs built from a public dataset that is increasingly incorporated with samples from the private dataset.

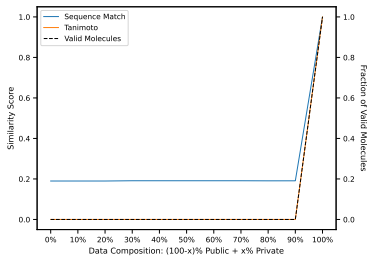


(c) Similarity scores for reagents encoded with a LUT that is built on the private training set and decoded with LUTs built from public data and increasingly updated with the correct LUT mappings ordered by frequency.

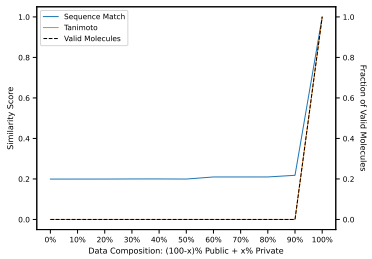


(d) Similarity scores for reagents encoded with a LUT that is built from the private training set and decoded with LUTs built from public data and increasingly updated with the correct LUT mappings in a random order.

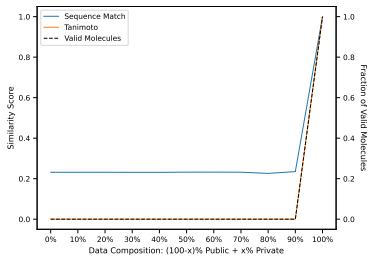
Figure 8: Privacy preserving results of reagents from the confidential chemical reactions dataset encoded with the correct LUT built from this dataset and decoded with imperfect LUTs. Patch size set to 4. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded reagent and groundtruth. The plot additionally depicts the fraction of invalid decoded reagents. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.



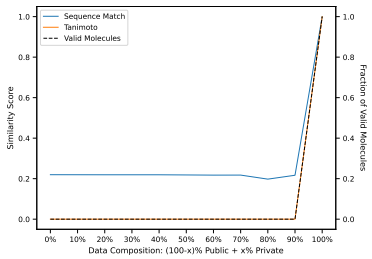
(a) Patch size = 3.



(b) Patch size = 5.

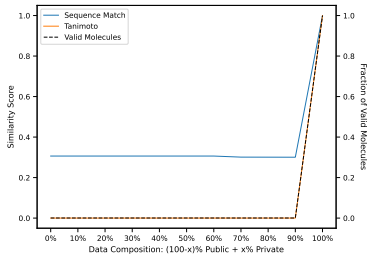


(c) Patch size = 10.

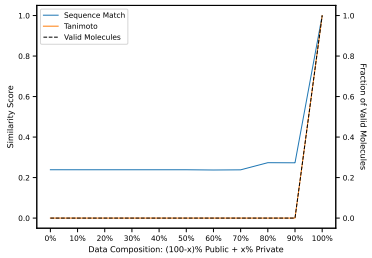


(d) Patch size = 20.

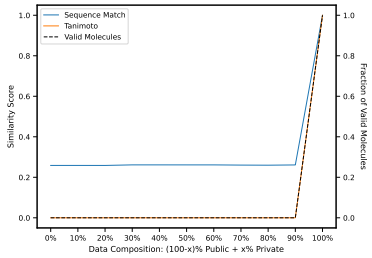
Figure 9: Sequence similarity scores of products from the confidential chemical reactions dataset encoded with the correct LUTs and decoded by LUTs built from a public dataset which is increasingly incorporated with samples from the private dataset for different patch sizes. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded product and groundtruth. The plot additionally depicts the fraction of invalid decoded products as a dashed line. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.



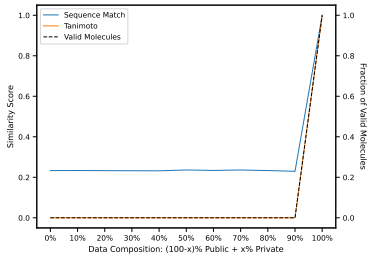
(a) Patch size = 3.



(b) Patch size = 5.

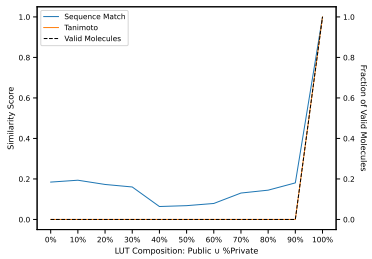


(c) Patch size = 10.

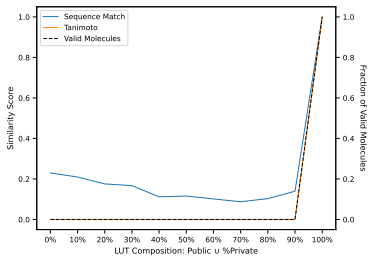


(d) Patch size = 20.

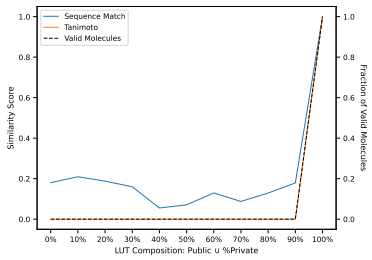
Figure 10: Sequence similarity scores of reagents from the confidential chemical reactions dataset encoded with the correct LUTs and decoded by LUTs built from a public dataset which is increasingly incorporated with samples from the private dataset for different patch sizes. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded reagent and groundtruth. The plot additionally depicts the fraction of invalid decoded reagents. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.



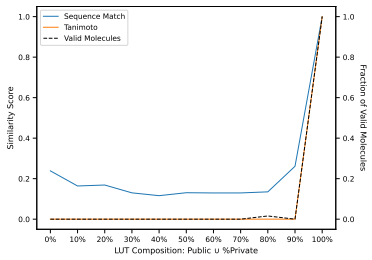
(a) Patch size = 3.



(b) Patch size = 5.

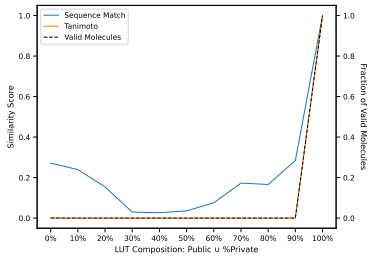


(c) Patch size = 10.

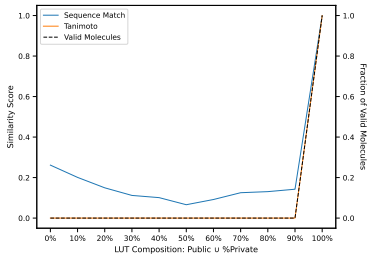


(d) Patch size = 20.

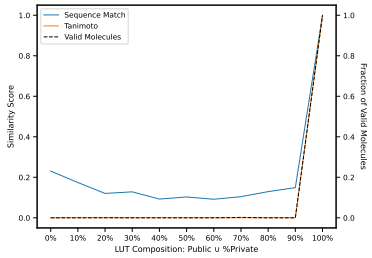
Figure 11: Sequence similarity scores of products from the confidential chemical reactions dataset encoded with the correct LUTs and decoded by LUTs built from public data and increasingly updated with the correct LUT mappings ordered by frequency. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded product and groundtruth. The plot additionally depicts the fraction of invalid decoded products as a dashed line. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.



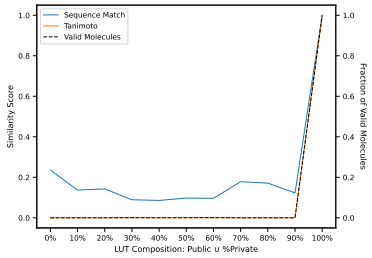
(a) Patch size = 3.



(b) Patch size = 5.

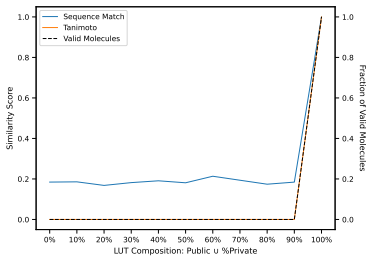


(c) Patch size = 10.

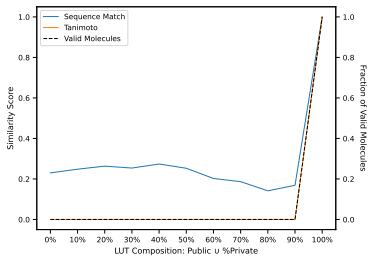


(d) Patch size = 20.

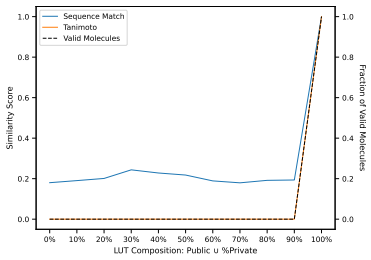
Figure 12: Sequence similarity scores of reagents from the confidential chemical reactions dataset encoded with the correct LUTs and decoded by LUTs built from public data and increasingly updated with the correct LUT mappings ordered by frequency. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded reagent and groundtruth. The plot additionally depicts the fraction of invalid decoded reagents as a dashed line. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.



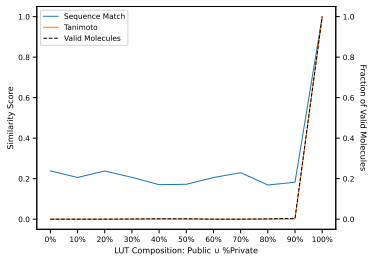
(a) Patch size = 3.



(b) Patch size = 5.

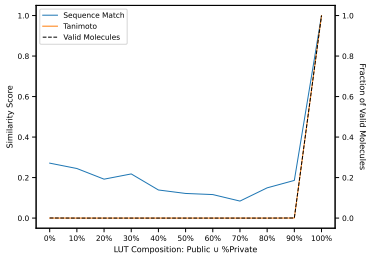


(c) Patch size = 10.

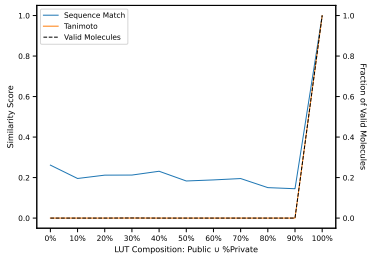


(d) Patch size = 20.

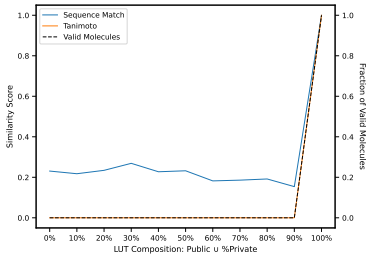
Figure 13: Sequence similarity scores of products from the confidential chemical reactions dataset encoded with the correct LUTs and decoded by LUTs built from public data and increasingly updated with the correct LUT mappings in a random order. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded product and groundtruth. The plot additionally depicts the fraction of invalid decoded products as a dashed line. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.



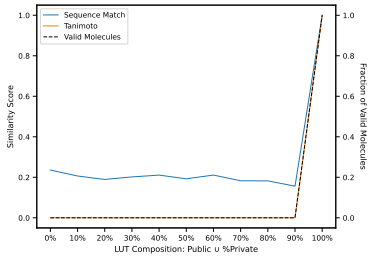
(a) Patch size = 3.



(b) Patch size = 4.



(c) Patch size = 10.



(d) Patch size = 20.

Figure 14: Sequence similarity scores of reagents from the confidential chemical reactions dataset encoded with the correct LUTs and decoded by LUTs built from public data and increasingly updated with the correct LUT mappings in a random order. Metrics are sequence match ratio (blue) and Tanimoto similarity (orange) of the decoded reagent and groundtruth. The plot additionally depicts the fraction of invalid decoded reagents as a dashed line. Note that the molecules are valid only when more than 90% of the private data is included in the public dataset.